Approche de Co-Simulation en vue de la validation et de l'évaluation de performance des systèmes de communication : Application à des architectures de distribution électriques

Mohamad Haffar

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : Automatique et productique

Arrêté ministériel : 7 août 2006

Présentée par

Mohamad Haffar

Thèse dirigée par **Pr. Jean Marc Thiriet**

préparée au sein du **Laboratoire GIPSA Lab**
dans **l'École Doctorale Electronique Electrotechnique Automatique et Traitement du Signal**

Approche de Co-Simulation en vue de la validation et de l'évaluation de performance des systèmes de communication : Application à des architectures de distribution électriques

Thèse soutenue publiquement le **« 21 Septembre 2011 »**,

devant le jury composé de :

M. Laurent CAUFFRIEZ

Maître de Conférences à l'Université de Valencienne et du Hainaut Cambrésis, Rapporteur

M. Thierry VAL

Professeur à l'Université de Toulouse, Rapporteur

M. Didier Georges

Professeur à l'INP-grenoble, Président

M. Jean-Philippe GEORGES

Maître de Conférences à l'Université Henri Poincaré Nancy 1, Examinateur

M. Jean-Marc THIRIET

Professeur à l'Université Joseph Fourier Grenoble, Directeur de thèse

M. Eric SAVARY

Directeur de l'Entreprise Euro System, Examinateur

A mon père Abdul Majid Haffar & Ma mère Yumna Korhani

Ma très chère femme Sarah Kabbara Haffar

Mon frère Walid Haffar

Remerciements

J'adresse mes remerciements et mes profondes gratitudes à mon directeur de thèse, Monsieur Jean Marc THIRIET, qui m'a permis grâce à sa confiance et ses conseils constructives de mener à terme ce travail de recherche et développement. Un grand merci pour ta confiance et pour tout ce que tu as fait pour moi.

Je tiens également à adresser même mes sincères reconnaissances à Monsieur Eric Savary, président de l'entreprise Euro System, qui m'a assuré tout au long de mon parcours professionnel l'ambiance nécessaire qui m'a permis d'accomplir ce travail de R&D.

J'exprime mes remerciements à mes deux rapporteurs M. Thierry Val de l'Université du Toulouse et M. Laurent Cauffriez de l'Université de Valencienne et du Hainaut Cambrésis, pour avoir accepté d'être rapporteurs de mes travaux de thèse ainsi que pour leur lecture minutieuse du manuscrit. Merci également à M. Jean Philippe Georges de l'Université Henri Poincaré Nancy 1 de vouloir accepter sa présence en tant qu'examinateur. Et finalement je remercie M. Didier Georges de m'avoir fait l'honneur de présider le jury de cette thèse.

Un grand merci pour mes deux chefs de projets de l'entreprise Euro System Mr. Philippe Zucchet et Mr Daniel Baudrand qui m'ont accompagné techniquement et moralement tout au long de mon parcours. Merci de même à M. Mohamad Nachar qui a démarré avec moi en tant qu'un stagiaire et m'a donné une grande aide au niveau de ce développement de recherche, et a intégré l'équipe en tant qu'ingénieur grâce à ces fortes compétences.

Je commence, pour les externes, par offrir mon éternel amour à ma femme Sarah Kabbara Haffar qui a vécu avec moi les jolis moments et m'a soutenu pendant les difficiles moments de ces années de travail. Je t'offrirai spécialement et sincèrement ce grand diplôme de Doctorat.

Je remercie aussi Dr. Maher El Rafei et futur Dr. Walaa Sahyoun qui étaient pour moi un vrai frère et une vraie sœur. Sans détailler les jolis moments que nous avons vécu ensemble je vous remercie tous les deux de tout mon cœur.

Un grand merci à Dr Hayssam Ziade qui était proche de moi dans beaucoup de situations difficiles en me donnant ses conseils constructives pour pouvoir s'y affranchir et qui m'a donné de même une opportunité pour réaliser un cours professionnel au sein de la faculté de génie l'université libanaise. Je remercie également à cette occasion Dr Khaled Mouchref.

Je tiens aussi à remercier mes amis Mohamad Barakat, Mohamad Achkar, Ahmad Achkar, Ahmad Habli, Amine Mechraoui, Zeeshan Khan, Ayman Trad, Mohamad Obayyan pour tous les jolis moments nous avons passé ensemble. Je transmets mes spéciaux remerciements à Bahaa Ghamraoui et à Ahmad Issa.

Finalement, un message que j'aimerai transmettre à mon père Abdul Majid pour lui dire que j'espère que tu es ou courant que je suis arrivé à cette position que tu avais planifié pour moi, je pense à toi pour toujours. Ma mère Yumna, tu étais toujours inquiète pour moi mais finalement j'ai pu y arriver ; j'espère que tu seras toujours fière de moi. Mon adorale frère Walid, ma sœur Nada et Dima je vous remercie aussi.

Table des matières

Liste des abréviations

ACSI	**A**bstract **C**ommunication **S**ervice **I**nterface
CID	**C**onfigured **I**ED **D**escription
DA	**D**ata **A**ttribute
DO	**D**ata **O**bject
DUT	**D**evice **U**nder **T**est
FAT	**F**actory **A**cceptance **T**est
GOOSE	**G**eneric **O**bject **O**riented **S**ubstation **E**vent
HITL	**H**ardware **I**n **T**he **L**oop
ICD	**I**ED **C**apability **D**escription
IEC	**I**nternational **E**lectrotechnical **C**omission
IEEE	**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers
IID	**I**nstantiated **I**ED **D**escription
LD	**L**ogical **D**evice
LN	**L**ogical **N**ode
MICS	**M**odel **I**mplementation **C**onformance **S**tatement
MMS	**M**essage **M**anufacturer **S**pecification*
MUT	**M**odel **U**nder **T**est
OpNet	**OP**timized **N**etwork **E**ngineering **T**ool
PICS	**P**rotocol **I**mplementation **C**onformance **S**tatement
PD	**P**hysical **D**evice
SAT	**S**ite **A**cceptance **T**est
SCL	**S**ubstation **C**onfiguration **L**anguage
SCD	**S**ubstation **C**onfiguration **D**escription
SCSM	**S**pecific **C**ommunication **S**ervice **M**apping
SDC	**S**ystème **D**e **C**ommunication
SDEE	**S**ystème de **D**istribution de l'**E**nergie **E**lectrique
SDF	**S**ûreté **D**e **F**onctionnement
SDP	**S**ystème **D**e **P**rotection
SITL	**S**oftware **I**n **T**he **L**oop
SSD	**S**ystem **S**pecification **D**escription
SV	**S**ampled **V**alue

Table des figures

Liste des publications

Publications Scientifiques

Haffar M., Thiriet J-M., Nachar M., A Hardware in the Loop module in an IEC61850 Co-Simulation Platform for advanced substation automation system tests, *in* 'IEEE International Energy Conference and Exhibition', EnergyCon 2010, 18-22 Décembre 2010, Manama, Bahrain, paper 1569328789.

Haffar M., Thiriet J-M, Kabbara S., Validation of Hardware and Software in the Loop add-ons simulation modules, *in* 'Journées de la Section Automatique Démonstrateurs en Automatique 30 novembre', 1 décembre 2010 à Angers.

Haffar M., Thiriet J-M, Software and hardware in the loop component for an IEC 61850 Co-Simulation platform, *in* 'Proceedings of the International Multiconference on Computer Science and Information Technology', pp. 817–823, 18-20 October 2010, Wisla, Poland. ISBN 978-83-60810-27-9 ISSN 1896-7094.

Haffar M., Thiriet J-M, Mechref K., Ziade H., Transforming your computer center to an Emulated Industrial Laboratory, *in* 4ème congrès international francophone de mécanique avancée, Alep (Syrie), 19-21 avril 2010.

Haffar M., Khan Z-H., Thiriet J-M, Savary E., An extension to IEC 61850 for solving selectivity problem in electrical substations, *in* '23rd IAR Workshop on Advanced Control and Diagnosis 2008', Coventry, 27-28 November 2008, pp . 362-367.

Haffar M., Thiriet J-M, Savary E., Modeling of substation architecture implementing IEC 61850 protocol and solving interlocking problems, *in* '7th IFAC International Conference on Fieldbuses & Networks in Industrial & Embedded systems', Toulouse, France (November 7-9, 2007), pp. 291-294.

Publications Industrielles

Haffar M., Savary E., Baudrand D., IEC61850 Co-Simulation platform: an advanced tool for Substation Automation System design architecture, *in* 'IEEE International Energy Conference and Exhibition', EnergyCon 2010, 18-22 Décembre 2010, Manama, Bahrain, paper 1569315305.

Tutoriels

Bruandet G., Angays P., **Haffar M.**, Saxena P-K. [2010], IEC 61850 Tutorial, *in* 'Petroleum and Chemical Industry Technical Conference', PCIC 2010 Europe.

L'intérêt de notre travail de recherche a fait l'objet d'une partie d'un cours industriel, d'une durée de deux heures, présenté sous la forme d'un tutoriel publié dans la conférence PCIC 2010 Europe (IEEE). Le but de ce cours était d'expliquer les points essentiels de la norme IEC 61850 et de montrer le rôle des différentes sociétés pour l'installation d'un système de distribution d'énergie électrique basé sur cette norme. Ce cours a été produit suite à une collaboration industrielle entre la société d'intégration système Euro System, les deux constructeurs des appareils Schneider Electric, Areva et la société d'ingénierie Technip. J'ai

représenté la société Euro System pour montrer le rôle des intégrateurs systèmes dans la chaine de construction, vérification et test d'un nouveau projet à base de ce nouveau standard de communication. Le nombre de participants à ce Tutoriels a atteint le palier de 70 personnes dont la majorité était des donneurs d'ordres et des chercheurs industriels.

Chapitre 1

Introduction générale

Un système de distribution électrique est le cœur de tous types de sites industriels, aussi bien les sites producteurs d'énergie que les sites consommateurs. Les équipements primaires constituant le système de distribution électrique sont des appareils ayant différents niveaux de tension tels que les groupes électriques, les transformateurs, les jeux de barres et les éléments de coupure. Ces appareils sont répartis de manière hiérarchique afin d'assurer la distribution de l'énergie fournie par les groupes électriques vers les différents récepteurs du site **{chapitre 2}**.

Afin d'assurer la protection des équipements primaires contre les défauts électriques pouvant apparaître dans un système de distribution, des unités basées sur des microcontrôleurs sont implémentées dans ce système pour assurer la fonction de surveillance des données électriques et la mise en sécurité du système global, le cas échéant. Ces unités, intitulées Systèmes De Protections 'SDP', agissent à deux niveaux **{chapitre 2}**:

- le premier niveau, local, permet l'interaction directe entre le SDP et l'équipement primaire,
- le deuxième niveau consiste en la coopération de plusieurs SDP afin de se prémunir contre des défaillances pouvant affecter plusieurs équipements primaires et aboutir dans certains cas au black out du système de distribution complet.

Le premier type est simple à mettre en œuvre et ne pose pas de problèmes particuliers. Par contre, le deuxième type de protections nécessite la prise en compte du problème de communication. Cet aspect est crucial car il a des conséquences directes sur la sûreté de fonctionnement du système de distribution d'énergie. La mise en œuvre n'est pas aisée car il est nécessaire de partager les fonctionnalités par plusieurs systèmes de protection.

Jusqu'à une période récente, pour des raisons de simplicité, de sûreté de fonctionnement et de disponibilité technologique, les entreprises du domaine et les intégrateurs utilisaient encore des solutions basées sur des liaisons point à point entre les SDP. Aujourd'hui, le problème de la complexité des architectures se pose ainsi que le besoin d'interopérabilité exprimé par les utilisateurs. Par conséquent, l'utilisation de réseaux de communication, couplée à des systèmes de supervision « standardisés », peut répondre à ce problème ; ceci permet de simplifier le nombre d'interconnexions câblées entre les SDP afin de réduire les coûts d'installation et d'augmenter la modularité et la flexibilité des SDP. L'utilisation d'un réseau sur ce type d'application pose le problème du niveau de sûreté de fonctionnement de la communication, en particulier il nous faudra la garantie de transmission des messages de sécurité dans un délai temporel borné **{chapitre 2}**.

L'année 2003 a vu la naissance d'une nouvelle norme intitulée 'IEC 61850', basée sur TCP/IP, dédiée aux systèmes de communication des installations de distribution électriques **{chapitre 3}**.

Cette norme préconise l'utilisation d'un seul réseau, avec une représentation virtuelle de différents nœuds logiques interconnectés. Ces nœuds logiques doivent être capables d'envoyer deux types de trafic dont chacun contient plusieurs services de communication :

- un trafic "normal" qui utilise les protocoles et procédures classiques de TCP/IP,
- un trafic "sécuritaire" (alarmes par exemple) qui doit être transmis coûte que coûte indépendamment des conditions d'utilisation du réseau et qui s'appuie sur les protocoles 802.1p/q.

Des exigences en termes de performance temporelle des messages sécuritaires sont imposées par la norme afin de vérifier la viabilité de ces messages et les rendre exploitable par les protections distribuées. La norme impose la présence d'une méthodologie permettant l'évaluation et les tests de performance des messages sécuritaires d'une telle architecture de communication mais ne donne aucune indication sur les outils ou les approches à utiliser.

Notre travail de recherche avait pour objectif principal d'identifier l'approche la plus adaptée pour accomplir cette tâche. Cette dernière doit permettre en phase de conception, de choisir l'architecture de communication et les flux de données les plus adaptés aux spécifications du système et aux exigences normatives, elle doit se poursuivre en phase de vérification et de tests, pour valider la fiabilité des messages de communication avant de passer en phase d'exploitation **{chapitre 3}**.

En conséquence, et pour les causes expliquées dans le chapitre 3, nous avons opté pour le développement d'une plateforme basée sur l'approche de Co-Simulation. Le **{chapitre 4}** aborde les différents modules développés au cours de notre étude qui ont permis de rendre opérationnelle cette plateforme de Co-Simulation.

Une première utilisation de cette plateforme a été effectuée sur un protocole de communication répandu dans le monde industriel (ModBus/IP), et ce dans le but de montrer l'intérêt qu'apporte cette plateforme en terme de validation de la conformité de l'implémentation des protocoles de communication dans les modèles de simulation **{chapitre 5}**.

Plusieurs Co-Simulations ont été de même effectuées dans le **{chapitre 6}** afin d'évaluer les performances des architectures de communication hybrides (ModBus/IP) qui implémentent à la fois les modèles créés dans la plateforme de Co-Simulation et des équipements industriels. L'analyse des résultats de performances obtenus dans ce chapitre nous a permis d' identifier les retards ajoutés par les modules de la plateforme sur les performances des Co-Simulations. Ces délais imposent des contraintes d'utilisation de cette plateforme.

La modélisation du standard de communication IEC 61850 a fait l'objet des derniers résultats obtenus au cours de nos travaux de recherche **{chapitre 7}**. Un des services essentiels du trafic normal TCP/IP de la norme, intitulé auto-découverte, a été implémenté dans nos premiers modèles. Ce service permet par une simple interrogation envoyée par des équipements de tests, de découvrir le contenu Objet des équipements ou des modèles conforme à la norme IEC 61850. La validation de la conformité de ce service modélisé est montrée à la fin de ce chapitre.

Chapitre 2

Sûreté de fonctionnement & Système de distribution d'énergie électrique

2.1 Introduction

Notre travail de recherche est centré autour de l'étude de la **S**ûreté **D**e **F**onctionnement (SDF) des automatismes distribués appliqués à des sous stations-électriques. Ces automatismes sont connus sous le nom de '**S**ubstation **A**utomation **S**ystem' [Apostolov et al, 2003], [Caetano et al, 2007] et [Santos et al, 2009]. Ces sous-stations font partie d'un **S**ystème global de **D**istribution d'**E**nergie **E**lectrique (SDEE) qui sera le cœur de notre étude. Le SDEE est un système qui permet la livraison de l'énergie électrique produite par les groupes de production ou les centrales électriques vers les consommateurs de l'installation électrique dans laquelle il se trouve. Divisé en plusieurs sous-stations électriques, ce système englobe plusieurs autres sous-systèmes interconnectés entre eux.

La première partie de ce chapitre, illustrée dans le paragraphe 2, montre les différentes caractéristiques de la SDF d'un système général d'automatisation. Les attributs, les moyens ainsi que les menaces liées à la SDF sont décrits dans cette partie.

Le paragraphe 3 présente dans la suite les différents sous-systèmes constituant un SDEE. Ce paragraphe met l'accent sur un sous-système particulier et critique, intitulé **S**ystème **D**e **P**rotection (SDP). Les applications d'automatismes permettant d'assurer la protection du SDEE sont présentées dans ce paragraphe. Ces automatismes sont soit implémentés en local au SDP soit distribués entre plusieurs SDP.

Le paragraphe 4 se focalise sur les applications les plus couramment utilisées en industrie, permettant la réalisation des automatismes distribués entre les SDP. Plusieurs approches existent pour la mise en place de ces automatismes, certaines sont basées sur un simple réglage des SDP et d'autres exigent des interconnexions entre ces systèmes réalisées soit par l'intermédiaire de liaisons point à point soit d'un réseau de communication. Ces différentes approches sont présentées dans ce paragraphe avec une vision historique.

Le dernier paragraphe synthétise l'influence des automatismes distribués sur les attributs de la SDF d'un SDEE. Ce paragraphe explique ainsi le lien entre les trois paragraphes essentiels de ce chapitre.

2.2 Architecture générale d'un SDEE

Un SDEE est composé de plusieurs sous systèmes interconnectés entre eux par l'intermédiaire des entrées/sorties physique ou par l'intermédiaire des réseaux de communication.

La partie principale constitue l'architecture électrique contenant les fournisseurs de l'énergie connectés aux récepteurs ou charges par le biais d'une structure électrique de distribution d'énergie (figure 2.1) {chapitre 2, paragraphe 2.4.2, paragraphe 2.4.3}.

Cette architecture électrique est surveillée par les SDP qui collectent les valeurs électriques associées à chaque composant de cette architecture par le biais des équipements de transformation (figure 2.1) et envoient les commandes à la suite des procédures de contrôles implémentées dans ces derniers.

En plus du contrôle et de la surveillance de l'architecture électrique, ces SDP assurent des échanges de communication par l'intermédiaire d'un Système De Communication SDC afin d'échanger des informations entre eux et avec les Systèmes De Supervision SDS (figure 2.1).

Deux types de SDP existent, les anciens intègrent une communication basée sur un média série et nécessitent la présence d'un concentrateur de donnée et convertisseur de protocoles pour leur permettre de se connecter au SDC et les nouveaux intègrent une interface de communication Ethernet leur permettant une connexion directe avec le SDC (figure 2.1) [Haffar et al, 2007].

Les échanges de communications via le SDC peuvent avoir différents aspects, notre étude va se concentrer sur les échanges entre les SDP et le SDC.

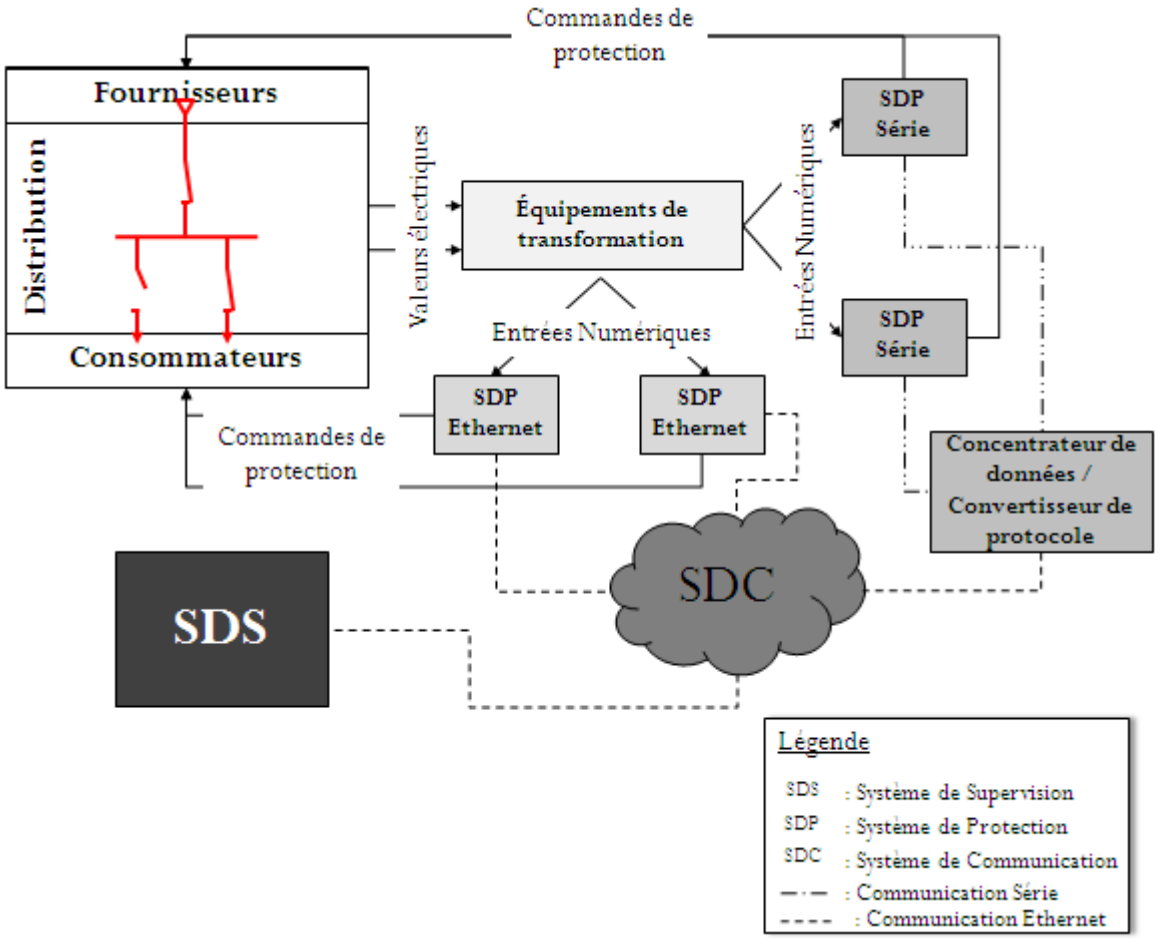


Figure 2.1 Architecture générale d'un système global de distribution d'énergie électrique

2.3 Concepts généraux de la SDF

Un système est une entité qui interagit avec d'autres systèmes aussi bien matériels que logiciels. L'ensemble des entités avec lesquelles le système interagit constitue son environnement. Les fonctions d'un système qualifient ses capacités. Elles sont décrites dans sa spécification fonctionnelle en termes de fonctionnalités et de performances [Avizienis et al, 2004]. La notion de service décrite dans [Laprie et al, 1995], précise le comportement du système vu par un utilisateur externe. Cet utilisateur peut être un autre système qui peut lui-

même fournir des services. L'échange d'un service entre un fournisseur et un utilisateur passe par l'intermédiaire d'une interface intitulée interface de service [Avizienis et al, 2004]. En prenant l'exemple d'un **S**ystème **D**e **C**ommunication (SDC), ce système est composé de plusieurs sous-systèmes intitulés composants qui communiquent entre eux. Les services de ces composants sont présentés sous la forme d'un échange des messages de communication au travers du réseau de communication liant ces différents composants. L'interface de service de chaque élément communicant est intitulée interface de communication.

N'importe quel système peut faire face à des défaillances le rendant incapable de réaliser les fonctions pour lesquelles il a été conçu. La SDF est la science des défaillances [Villemeur et al, 1988] qui permet l'analyse et l'évaluation des comportements d'un système face à des situations pouvant impacter son fonctionnement.

[Villemeur et al, 1988] résume le fait que, pour améliorer la fiabilité d'un système, il faut tenir compte non seulement des composants élémentaires de ce système mais aussi de l'interaction entre ces composants.

Par ailleurs, l'étude de la SDF devra être accomplie en partant des composants élémentaires du système pour ensuite étudier l'ensemble des interactions entre les différents éléments du système. Cette étude a fait l'objet de nombreuses publications dans diverses conférences [Avizienis et al, 2004] [Laprie et al, 1995] [Trivedi et al, 2009].

D'après [Avizienis et al, 2004], il existe deux définitions différentes de la SDF :

- la première définition consiste à justifier la SDF pour les services fournis par un système. Cette notion exige la mise en œuvre des moyens pour vérifier la capacité du système à délivrer des services sans failles.
- la deuxième définition exprime l'aptitude d'un système à éviter la défaillance de ces services. Ces défaillances peuvent en effet porter atteinte aux services fournis par le système. Dans ce cas, la SDF exige la définition des critères de bon fonctionnement des services critiques de ce système.

[Laprie et al, 1995] définit la SDF des systèmes informatiques comme étant la confiance accordée aux services délivrés aux utilisateurs du système.

Pour faciliter l'évaluation de la SDF, plusieurs publications [IEC TC56 WG3 et al, 2009] [Trivedi et al, 2009] ont préconisé l'utilisation de plusieurs attributs : la disponibilité, la confidentialité, l'intégrité, la performance, la fiabilité, la sécurité et la maintenabilité.

La confidentialité représente l'aptitude d'un système à ne pas divulguer des informations confidentielles à des parties non autorisées à les recevoir.

L'intégrité symbolise l'aptitude d'un système à éviter les changements/suppressions d'informations sans autorisation. Cet attribut permet de garantir la validité des données.

La performance exprime le degré pour lequel un système peut accomplir les fonctionnalités requises dans des contraintes bien spécifiques. Dans la littérature classique [Avizienis et al, 2004], ce paramètre n'est pas considéré comme un attribut essentiel de la SDF mais comme un élément impactant les attributs de cette dernière.

Notre étude se concentre sur l'influence du SDC sur la SDF d'un système global 'SDEE' expliqué dans {chapitre 2, paragraphe 3}. La performance définie auparavant est vue comme étant un attribut essentiel pour ce SDC. Néanmoins, du point de vue du système global SDEE, la performance est considérée comme un élément permettant de donner des indications sur les autres attributs du système global. D'autre part, l'intégrité est un attribut essentiellement utilisé dans des systèmes d'information, pour un SDEE cet attribut n'est pas pris en charge.

Finalement, l'attribut de confidentialité est essentiellement utilisé dans le domaine de la sécurité de réseau, pour cela, il est ignoré également dans notre étude.

Par conséquent, les attributs qu'on a retenus sont: la fiabilité, la maintenabilité, la disponibilité et la sécurité. Ces paramètres permettent de définir les objectifs attendus du SDEE et/ou d'évaluer la qualité de ces services afin de déterminer les points critiques pouvant affecter ces performances.

Le paragraphe suivant illustre brièvement la définition de ces différents attributs.

2.3.1 La fiabilité (Reliability)

La fiabilité est définie comme étant l'aptitude d'un système à assurer une continuité de ses services (i.e. ne pas défaillir) pendant une durée donnée (i.e. la durée de sa mission).

Elle est calculée par la probabilité du système à accomplir sa fonction requise pendant un intervalle de temps donné. Le système est considéré comme étant en fonctionnement normal à l'instant initial.

$R(t) = P$ {système fonctionne [0, t [}

Se limitant à un seul composant du système, [Yu et al, 1983] calcule la fiabilité d'un élément unitaire qui est évaluée par une fonction exponentielle de la forme :

$R(t) = e^{-\lambda t}$

λ: le taux de défaillance de l'unité défini comme étant l'inverse du temps moyen de défaillance.

La fiabilité de l'équipement est à son maximum à l'instant t=0 correspondant au temps de mise en service de l'équipement. Elle tend vers la valeur 1/e (~0,36) lorsque le temps de fonctionnement atteint le temps moyen de défaillance.

2.3.2 La maintenabilité (Maintainability)

La maintenabilité est l'aptitude d'un système à être modifié et/ou réparé pour rester dans un état de fonctionnement normal.

Elle est calculée par la probabilité que ce système soit réparé tout en étant en panne à l'instant t = 0 :

$M(t) = P$ {système est réparé sur [0, t [}

2.3.3 La disponibilité (Availability)

La disponibilité est l'aptitude d'un système à être prêt à répondre aux sollicitations des utilisateurs (même si la réponse envoyée peut être défaillante).

La disponibilité symbolisée par le paramètre 'A : Availability' sera calculée par la formule :

$A = MUT / (MUT+MDT)$

MUT : 'Mean Up Time' étant le temps moyen de fonctionnement et MDT : 'Mean Down Time' le temps moyen de dysfonctionnement.

2.3.4 La sécurité- innocuité (safety)

La sécurité peut être perçue selon 3 axes différents définis dans :

l'innocuité, la confidentialité, et l'intégrité.

Dans notre cas, nous allons nous intéresser à la sécurité dans le sens de l'innocuité qui est définie comme étant l'aptitude d'un système à ne pas générer des risques ou évènements critiques/catastrophiques sur les personnes et/ou les biens et/ou l'environnement.

2.3.5 Relations entre les différents attributs de la SDF

Les auteurs de la SDF considèrent que la disponibilité d'un système dépend entièrement de sa fiabilité et de sa maintenabilité. Toutefois, la sécurité n'est que partiellement dépendante de ces attributs.

En effet, les défaillances affectent la fiabilité du système et le rendent indisponible. D'autre part, un manque de maintenabilité du système peut introduire une augmentation du nombre de défaillances, et donc diminuer sa disponibilité.

En outre, l'occurrence d'une défaillance (i.e. fiabilité) peut engendrer des accidents, et en plus les opérations de maintenance peuvent exposer le système à des risques d'accidents. La sécurité d'un système dépendra donc de sa fiabilité et de sa maintenabilité.

En revanche, contrairement à la sécurité, la disponibilité d'un système dépend entièrement de sa maintenabilité et de sa fiabilité. En effet, les risques peuvent apparaître dans un système sans l'introduction d'une défaillance et hors la phase de maintenance.

La figure 2.2 illustre la relation entre les différents attributs de la SDF d'un système.

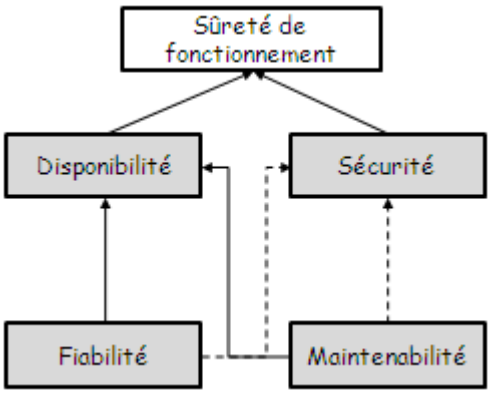


Figure 2.2 Relation entre les attributs de la SDF

La spécification fonctionnelle doit décrire les exigences relatives aux attributs de la SDF d'un système. Selon les systèmes étudiés, les attributs pris en compte peuvent être différents et certains attributs peuvent ne pas être pris en compte dans la spécification.

2.3.6 Les menaces de la sûreté et de la sécurité

Une faute sur un système est définie comme étant l'évènement permettant de mettre les états internes de ce système en erreur.

Deux sources peuvent être à l'origine d'une faute :

- soit elle est produite en interne au composant et génère une erreur après traitement par le composant;
- soit elle est issue d'un autre composant, cette faute génère alors une erreur sur le composant (exemple des signaux de communication GOOSE-IEC 61850 {chapitre 3, paragraphe 3.2.4})

Le traitement de cette erreur peut générer, en fonction de sa criticité, un phénomène de propagation et peut atteindre l'interface de service du composant et activer un service vers les autres composants.

La défaillance de service est un évènement qui aura lieu lorsque ce service ne remplit plus les fonctionnalités requises. Le premier évènement qui peut déclencher une défaillance est le fait que les performances du service ne sont pas cohérentes avec celles décrites dans la spécification fonctionnelle du système. Elle peut aussi apparaître dans le cas où la spécification fonctionnelle n'est pas en adéquation avec les fonctions du système.

Le retour d'un service à l'état opérationnel est intitulé 'restauration du service'. En général, la défaillance se traduit par la transmission de l'erreur d'un composant vers un ou plusieurs autres composants du système.

La figure 2.3 illustre le lien entre les erreurs et les fautes ainsi que leur principe de propagation.

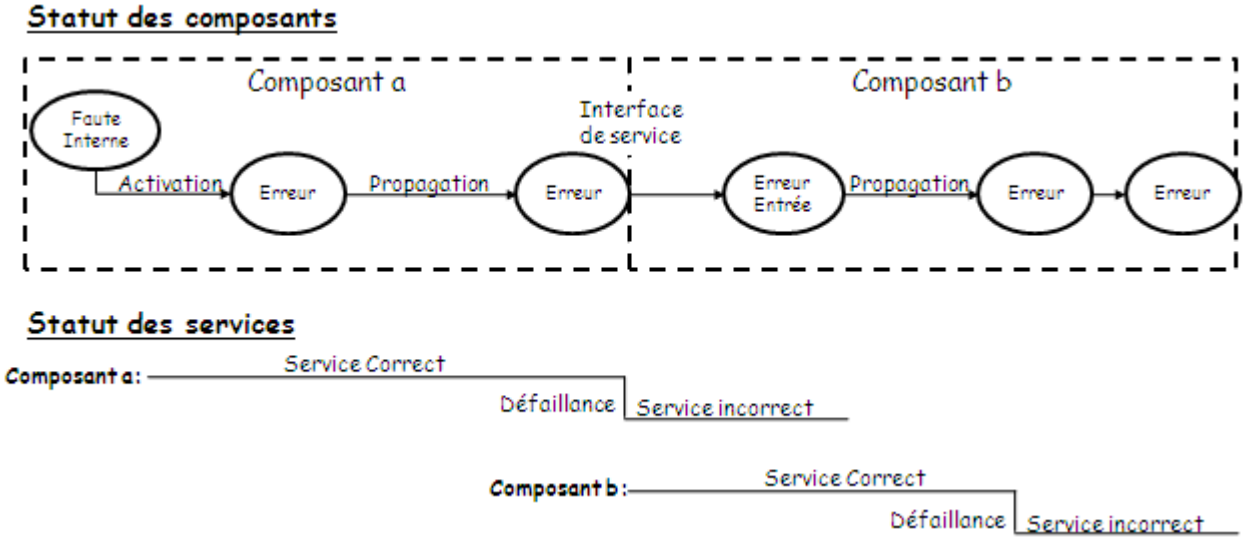


Figure 2.3 Relation de propagation des erreurs, fautes et défaillances

Dans certains cas, la défaillance d'une partie des services met le système dans un mode dégradé lui permettant de continuer à fournir uniquement les services non défaillants.

Selon [Avizienis et al, 2004], une défaillance est caractérisée par les points suivants:

- Domaine de la défaillance

- Détectabilité de la défaillance
- Conséquence de la défaillance

Pour mieux comprendre ces trois caractéristiques, prenons l'exemple d'un SDC.

Dans ce système, le domaine de la défaillance peut être divisé en deux catégories :

- la défaillance de contenu
- la défaillance temporelle

La première catégorie signifie que l'information délivrée par l'interface de service (i.e. l'interface de communication) n'est pas conforme au contenu défini dans la spécification fonctionnelle de ce service.

La deuxième catégorie signifie que la durée ou le temps mis pour accomplir ce service n'est pas conforme avec l'exigence temporelle définie dans la spécification fonctionnelle.

La détectabilité de défaillance est caractérisée par un signal émis par le système vers l'utilisateur.

Cette détectabilité peut elle-même être défaillante dans les deux cas suivants:

- un signal erroné envoyé à l'utilisateur en absence d'une défaillance sur le système
- aucun signal envoyé à l'utilisateur malgré la présence d'une défaillance sur le système

Les conséquences de défaillance produisent plusieurs niveaux de sévérités sur le système en allant d'une sévérité mineure jusqu'à une sévérité catastrophique. Chacun des niveaux est associé à un taux d'acceptation. La sévérité d'une défaillance influe directement sur la SDF du système.

2.3.7 Les moyens permettant d'atteindre un niveau de SDF

L'identification des causes des défaillances, le calcul des probabilités d'occurrence des fautes et l'évaluation des menaces de défaillance sont des caractéristiques liées à la sûreté des systèmes. Différents moyens existent dans le but d'évaluer ces caractéristiques. Ces moyens s'inscrivent dans l'ingénierie système et doivent être répartis tout au long des phases de développement d'un nouveau système : la spécification, la conception en passant par la réalisation, les tests usine, la mise en service et l'exploitation.

[Beugin Julie, 2006] précisent trois moyens pour atteindre la SDF :

- la tolérance aux fautes
- l'évitement des fautes
- la prévision des fautes

Notre analyse de ces moyens est définie de la manière suivante :

La tolérance aux fautes est un moyen qui permet d'assurer l'aptitude d'un système à fournir des services sûrs de fonctionnement même en cas d'apparition des fautes sur le système.

L'évitement et la prévision des fautes sont deux moyens qui permettent de fournir une confiance préalable à la SDF. Cette confiance est apportée en justifiant que le fonctionnel

d'un système et les spécifications de la sûreté sont bien adéquats et que le système peut bien les atteindre.

2.3.7.1 La tolérance aux fautes

C'est le moyen qui permet d'assurer une continuité de fourniture des services même si le système est en mode dégradé (i.e. malgré les fautes qui se produisent sur le système). La redondance [Hossenlopp et al, 2005], matérielle ou analytique, constitue l'une des techniques possible pour la conception d'un système tolérant aux fautes. Cette technique consiste à remplacer les services défaillants du système principal par des services appartenant à un système redondant.

2.3.7.2 L'évitement des fautes

Ce moyen consiste à réduire le nombre des fautes pouvant affecter un système en appliquant des approches de prévention sur le comportement de ce dernier. La prévention est généralement conduite pendant la phase de conception du système afin d'éviter l'occurrence ou l'introduction des fautes avant de le mettre en état d'opération. Elle est divisée en trois phases distinctes:

- la vérification
- le diagnostic
- la correction

La vérification permet d'examiner l'adéquation du système à ses propriétés. Dans le cas contraire, les deux phases de diagnostic et de correction entrent en jeu. La vérification sera nommée validation lorsqu'elle consiste à comparer la fonctionnalité d'un système avec sa spécification.

La non-satisfaction d'une spécification aura lieu lorsque des évidences peuvent être données montrant que le système n'implémente pas les fonctions demandées ou lorsque les fonctions implémentées ne peuvent pas être établies dans les conditions définies. La non-satisfaction pourra affecter n'importe quelle phase de développement d'un système (i.e. Système en phase de design, Système en phase de test usine, Système en phase de test sur site {chapitre 3, paragraphe 5}).

La technique de vérification peut s'effectuer dans deux modes :

- mode statique
- mode dynamique

2.3.7.2.1 Vérification statique

Ce mode consiste à évaluer les fonctionnalités d'un système qui ne sont pas affectées par son aspect opérationnel (i.e. le système ne doit pas être en mode dynamique). Ce mode pourra être appliqué soit sur le système lui-même sous la forme d'une analyse statique (e.g. analyse des flux de données) soit sur un modèle du système généralement établi par l'intermédiaire d'un graphe étape/transition (Réseaux de pétri, états finis, etc.).

2.3.7.2.2 Vérification dynamique

Ce mode consiste à évaluer les fonctionnalités d'un système qui sont impactées par son état opérationnel (i.e. dynamique), par exemple le test du système. Lorsqu'on effectue ces tests, le

système doit être dans un mode de fonctionnement particulier permettant d'accepter l'injection des entrées de tests dans le but d'évaluer le rapport entre le fonctionnel et la spécification du système.

Les entrées d'injection peuvent avoir à leur tour deux caractères différents :

- déterministe
- aléatoire

Les entrées ayant le caractère déterministe sont prédéterminées par un choix de sélection bien défini [Irith et al, 1993] [Cockburn et al, 1995]. Les tests à caractère aléatoire ou statistique suivent une distribution de probabilité définie généralement par des experts du système sur le domaine des entrées [Vaurio et al, 2002].

Lors d'une installation d'un nouveau système, des tests doivent être fournis tout au long des phases de développement du projet. Deux phases bien spécifiques existent généralement dans le but de vérifier et tester les fonctionnalités d'un système :

- la phase de test usine intitulé en anglais **F**actory **A**cceptance **T**ests **'FAT'**
- la phase de test sur site intitulé en anglais **S**ite **A**cceptance **T**est **'SAT'**

La phase de FAT se déroule tout d'abord, cette phase est effectuée généralement à l'endroit où le système est conçu. La phase de SAT se déroule quant à elle sur le site final dans lequel le système sera mis en service. Un test exhaustif conduit en phase de FAT permet de faciliter et de réduire les risques lors de la phase de mise en service du système. Ainsi, plus les tests sont sophistiqués et exhaustifs en FAT plus le temps de mise en service sera court. Ce point est considéré comme une demande essentielle pour la plupart des constructeurs [Bruandet et al, 2010].

Un test exhaustif d'un système est défini comme étant le test prenant en compte toutes les entrées possibles pouvant apparaître sur le système. Pour atteindre cette exhaustivité, il faut bien comprendre le principe de génération des entrées de tests qui peut être divisé en deux étapes :

- le choix des critères de la sélection des entrées de tests
- la génération des entrées de tests

Un test exhaustif est pratiquement inatteignable en raison du nombre illimité pour le choix des entrées de tests.

La figure 2.4 permet d'illustrer les différentes étapes du principe de vérification expliqué au cours de ce paragraphe :

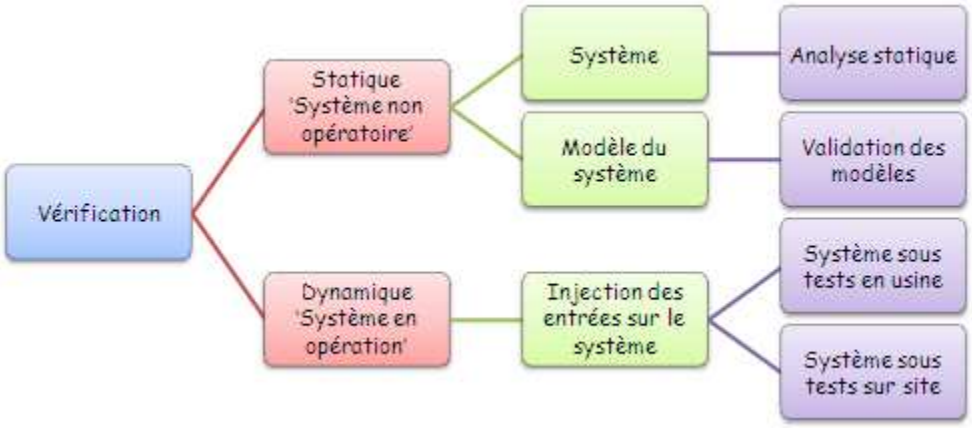


Figure 2.4 Technique de vérification

Le principe de l'élimination permet la correction des fautes par des techniques de diagnostic soit après la génération d'une erreur sur le système (mode correctif) soit par la prévention de la faute avant la génération d'une erreur (mode préventif). Ces modes peuvent être maintenus sans ou avec arrêt total du système.

2.3.7.3 La prévision des fautes

L'approche de prévision des fautes permet d'estimer la possibilité d'occurrence des fautes avant la conception d'un système. Ce moyen est généralement utilisé au niveau des phases initiales permettant la définition de la spécification fonctionnelle d'un nouveau système (i.e. avant sa conception et sa mise en service) [Avizienis et al, 2004].

Cette estimation peut être réalisée selon deux aspects (figure 2.5):

- quantitatif : identification et classification des modes de défaillances sans l'intervention d'un appel à des structures et des probabilités,
- qualitatif/probabiliste qui s'appuie sur des analyses structurelles et fonctionnelles permettant de déterminer les situations critiques pouvant affecter le fonctionnement d'un système [Beugin Julie, 2006]. Cette approche consiste à identifier les contraintes qui doivent s'appliquer à l'intégration des fonctions dans le but de respecter les attributs de la SDF du système.

En utilisant cette approche, une évaluation des attributs de la sûreté par l'intermédiaire des mesures physiques sera fournie. L'aspect qualitatif est généralement divisé en deux étapes :

- la modélisation (analytique ou de simulation)
- l'évaluation (Testing)

A son tour, la modélisation d'un système est divisée en trois phases : la construction des modèles élémentaires qui caractérisent le comportement de chaque composant constitutif du système, la validation de ces modèles élémentaires et finalement leur assemblage dans un environnement virtuel dans le but d'obtenir une image du système réel. Cette image permettra de donner des mesures relatives aux attributs de la SDF du système réel.

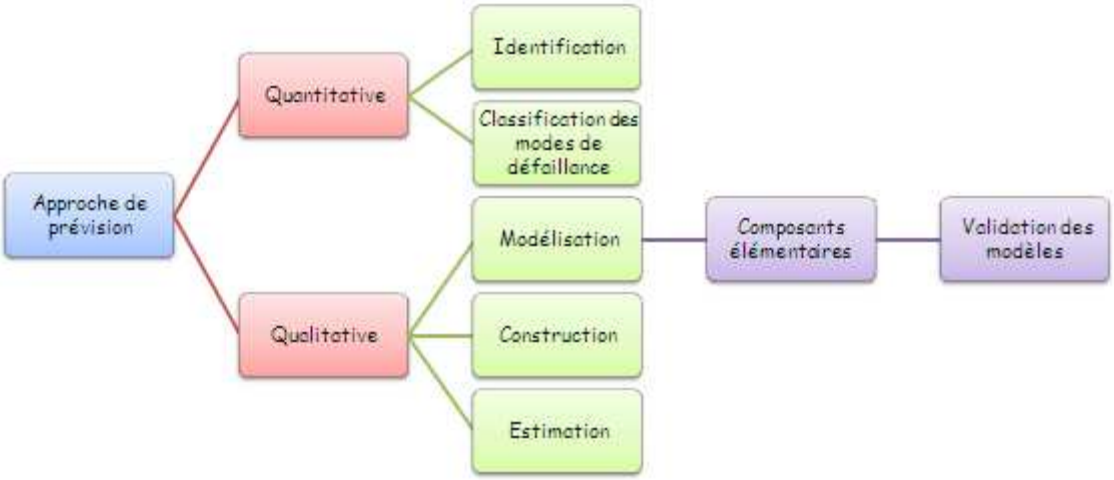


Figure 2.5 Arbre relatif à l'approche de prévision

2.3.8 Récapitulatif des caractéristiques de la SDF

Pour récapituler, les caractéristiques essentielles de la SDF d'un système sont:

- attributs
- menaces
- moyens

La figure 2.6 illustre le contenu les différents constituants de chacune de ces caractéristiques.

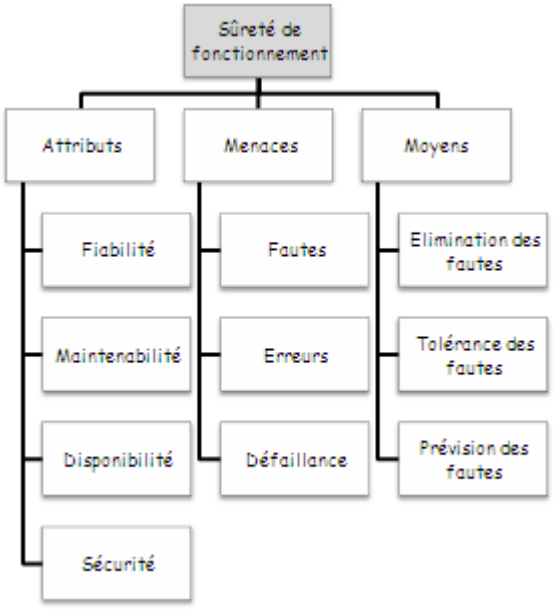


Figure 2.6 Caractéristique de la SDF

2.3.9 Quelques applications sur la SDF

Plusieurs études et applications ont été développées ces dernières années en tenant compte du respect de la SDF. Ce paragraphe va lister quelques références des études réalisées pour le calcul et l'évaluation des attributs de la sûreté en mettant l'accent en particulier sur celles qui sont liées à la SDF des applications de distribution d'énergie.

[Yu et al, 1983] pointe l'étude de la fiabilité et de la disponibilité sur les systèmes de distribution électrique. Selon les auteurs, le point d'entrée de calcul de la fiabilité d'un tel système revient à déterminer le taux de défaillance de chaque composant constituant le système. En fonction de la répartition structurelle de ces composants (architecture parallèle/série) un calcul des attributs de la sûreté sera ensuite effectué.

[Ito et al, 2008] prend en compte la SDF du réseau de communication impliqué dans une architecture de distribution d'énergie électrique. Les auteurs montrent qu'une amélioration de la disponibilité du système de communication est obtenue dans le cas d'une redondance des commutateurs réseaux et des serveurs de supervision.

2.4 SDEE et gestion électrique

2.4.1 Introduction

Un SDEE est un système qui regroupe des appareils permettant la production, la distribution et la consommation de l'énergie électrique. Son but essentiel est d'assurer une distribution fiable de cette énergie jusqu'aux récepteurs du site en subissant des adaptations des grandeurs électriques par l'intermédiaire de transformateurs. L'établissement de cette distribution nécessite la présence d'équipements spécifiques tels que les éléments de coupures, les jeux de barres, et les câbles.

Afin d'assurer une continuité de service, le SDEE est conçu de manière hiérarchisée sur plusieurs niveaux afin de limiter les conséquences d'un défaut en permettant d'isoler les circuits en cause, sans perturber le système globalement. Cet aspect permet aussi de faciliter les procédures de diagnostic et les opérations de maintenance.

2.4.2 Structure électrique générale

Tout d'abord, il est important de préciser que plusieurs seuils de tension peuvent être utilisés dans les architectures de distribution d'énergie. Ces seuils sont explicités ci-dessous dans la nomenclature correspondante à la norme française UTE C 18-510.

- HTB : Pour les tensions composées supérieures à 50 kV
- HTA : Pour les tensions composées comprises entre 1kV et 50 kV
- BTB : Pour les tensions composées comprises entre 500 V et 1 kV
- BTA : Pour les tensions composées comprises entre 50 V et 500 V
- TBT : Pour les tensions composées inférieures à 50 V

La structure générale montrée sur la figure 2.7 est composée de plusieurs niveaux:

- poste de livraison,
- poste général de distribution moyenne tension,
- réseau de distribution moyenne tension,
- tableaux et réseau de distribution basse tension.

Selon les exigences en termes de disponibilité requise d'alimentation électrique, plusieurs structures électriques peuvent être configurées pour chaque niveau de l'installation [Guide de conception des réseaux électriques industriel].

La figure 2.7 montre la structure générale d'une installation de distribution électrique.

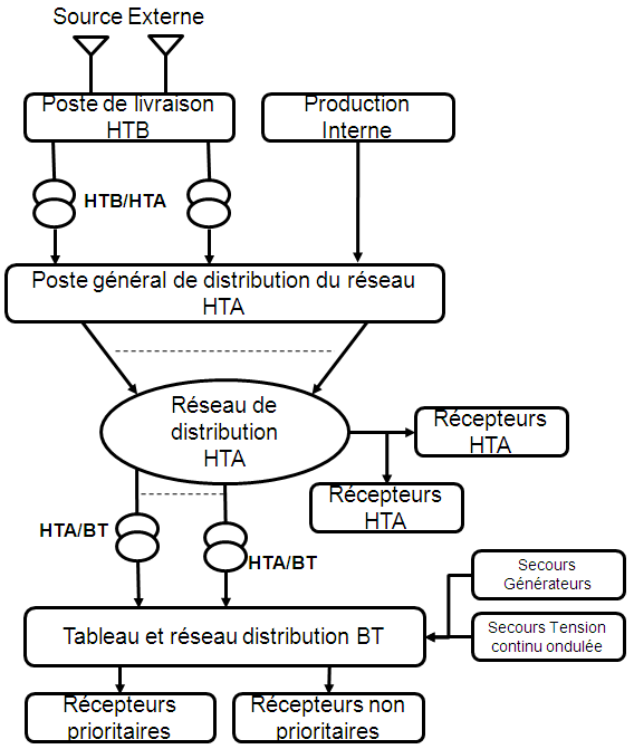


Figure 2.7 Structure générale d'un SDEE

Les arrivées électriques sont soit des sources externes au site (i.e. centrale électrique) soit des sources de production internes au site industriel (i.e. groupe électrogène). Seules les arrivées externes sont considérées comme entrées au niveau du poste de livraison. A la sortie de ce poste, un niveau de moyenne tension est délivré pour le réseau HTA.

Le poste général de distribution permet de dupliquer les arrivées moyenne tension pour les distribuer vers le réseau moyenne tension. Les sources de production internes s'additionnent à ce stade avec les départs moyenne tension du poste de livraison pour former les arrivées du poste général de distribution HTA. Les sources de production internes assurent une alimentation de secours dans le cas d'une perte des sources principales.

La structuration du réseau HTA est réalisée en respectant les contraintes de disponibilité, d'extensibilité et de coût. Des récepteurs HTA et des transformateurs HTA/BT sont raccordés à ce réseau dans le but d'assurer une alimentation des différentes charges moyenne tension et des départs électriques pour le réseau de distribution basse tension.

Outre le changement des niveaux de grandeurs électriques, les différents niveaux d'un SDEE se caractérisent par les nombres d'entrées (i.e. arrivées) et de sorties (i.e. départs) :

- le poste de livraison possède deux arrivées et deux départs électriques
- le poste général de distribution possède 3 à 4 arrivées, deux représentent les sources d'alimentation externes et les autres constituent les sources de production internes. Le nombre de départs est remarquablement plus élevé à cet emplacement de l'installation

dans le but d'assurer la distribution électrique à tous les bâtiments et les récepteurs de l'installation.

Finalement, le réseau électrique basse tension reçoit les départs transformés du réseau HTA et intègre plusieurs structures plus ou moins secourues par des générateurs locaux pour l'alimentation des charges électriques du site.

2.4.3 Structures de distribution électrique typiques

La figure 2.8 montre les trois architectures électriques les plus répandues dans un SDEE. Ces architectures sont données ici essentiellement à titre informatif pour préciser un peu le contexte.

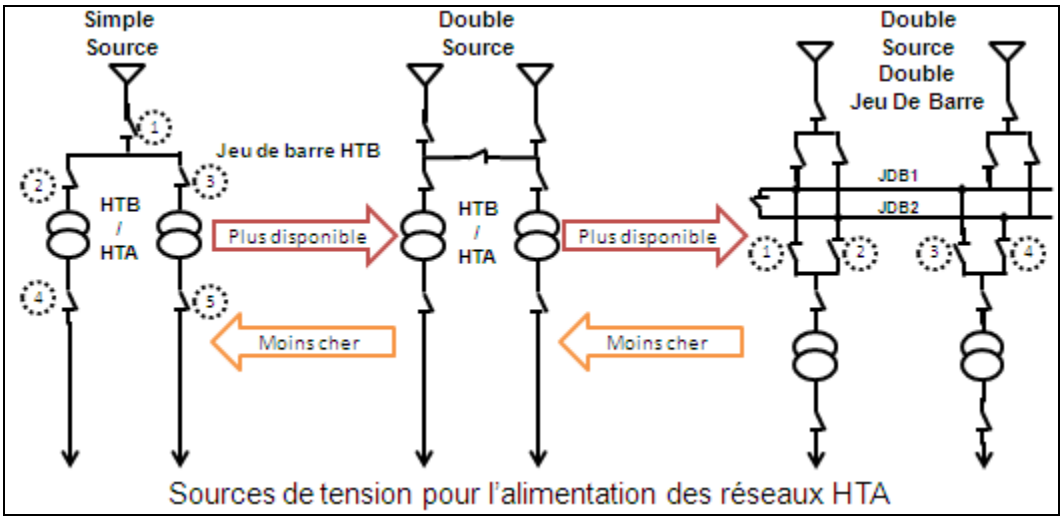


Figure 2.8 Architectures électriques typiques dans un SDEE

L'architecture **simple source** est basée sur une structure constituée d'une simple source d'alimentation connectée à un seul jeu de barre. Cette structure présente le coût optimal, en revanche elle dispose de la plus faible disponibilité en la comparant avec les autres architectures.

L'architecture **double source** dispose d'une structure électrique composée de deux sources d'alimentation connectées à un seul jeu de barre. Les éléments de coupure liés à cette structure sont normalement fermés de manière à assurer une alimentation continue même en cas de perte d'une des deux sources. Néanmoins, la maintenance des jeux de barres d'une telle structure entraînera un fonctionnement partiel de celle-ci.

L'architecture **double source avec double jeu de barre** présente une redondance au niveau des jeux de barres et des sources d'alimentation. Une des configurations initiales est produite de manière à alimenter chaque jeu de barre par chacune des sources. La maintenance ou la perte d'un jeu de barre ou d'une source n'entraînera aucune indisponibilité d'alimentation électrique. L'inconvénient d'une telle solution est qu'elle présente un coût d'installation important.

Ces trois structures de base peuvent apparaître à n'importe quel niveau du SDEE, en particulier au niveau du poste de livraison, poste général de distribution ou même au niveau des réseaux de distribution moyenne et/ou basse tension.

Le réseau de distribution moyenne tension est généralement divisé en plusieurs sous-stations électrique ayant chacune 2 arrivées venant du poste général de distribution et plusieurs départs qui sont les tableaux moyenne tension. Des structures électriques additionnelles peuvent apparaître dans ce type de réseau dans le but d'ajouter de la souplesse à l'installation. Pour un réseau étendu pouvant engendrer des extensions futures, une structure de type bouclé sera par exemple choisie car elle autorise la mise en place d'extensions [Guide de conception des réseaux électriques industriel].

Finalement, une structure à base de triple alimentation avec ou sans couplage peut être choisie pour les tableaux basses tensions. L'augmentation du nombre de sources d'alimentation (redondance) assure un accroissement de la disponibilité vis-à-vis de la perte d'une source. La différence entre un système avec couplage et un système sans couplage est la possibilité de réagir suite à une demande excessive de charge.

La figure 2.9 montre un exemple d'architecture basse tension composée de deux tableaux BT. Les deux tableaux sont interconnectés électriquement entre eux et possèdent chacun une structure électrique spécifique. Cette architecture sera prise comme architecture de référence du fait que les différentes protections à base d'automatismes distribués, expliquées dans le paragraphe 2.5, peuvent être appliquées sur cette architecture.

Le tableau 1 est alimenté avec une structure en double source avec couplage tandis que la structure électrique du tableau 2 est en mode triple source d'alimentation sans couplage. Une des sources du tableau 2 est issue du tableau 1 d'où l'interconnexion entre les deux tableaux. Le mode normal de fonctionnement d'une telle architecture est traduit par des contacts normalement fermés au niveau des éléments de coupure 1 et 3 et 4 et des contacts normalement ouverts au niveau des éléments de coupures aux emplacements 2 et 5. En cas de perte d'alimentation ou d'un défaut/maintenance de jeu de barre alimentant le tableau 2 une mise en mode secours sera effectuée en basculant les états des éléments de coupure aux points 4 et 5.

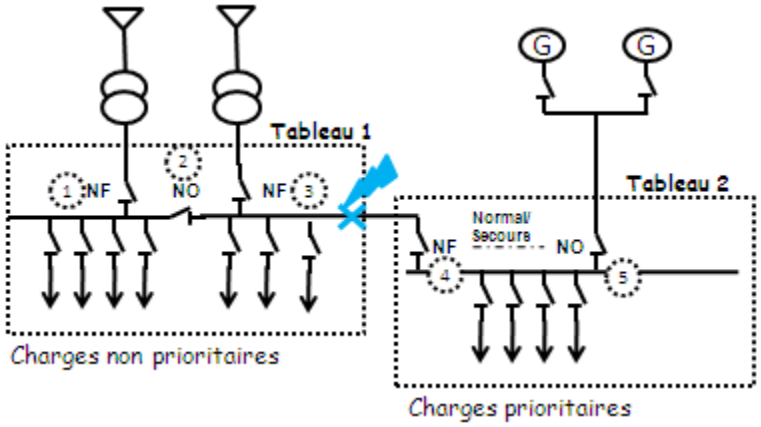


Figure 2.9 Interconnexion entre deux tableaux basses tension de structures différentes

2.4.4 Les systèmes de protection

Dans le but de détecter des défauts électriques pouvant éventuellement se produire dans un SDEE, chaque équipement (Jeu de barre, transformateur, source de production, élément de coupure, etc.…) doit être contrôlé et commandé par une unité suffisamment intelligente lui

assurant sa protection. Cette unité est appelée SDP. Le SDP sera ainsi considéré comme un système global embarquant plusieurs autres systèmes.

En prenant l'exemple d'un court circuit se produisant dans le tableau 1 de la figure 2.10, le SDP associé à l'élément de coupure doit détecter ce défaut lors de son apparition. Par la suite, ce système doit lancer la procédure de contrôle dans le but de commander le déclenchement du contact de l'élément de coupure en question.

Ce SDP contribue entre autre à l'élimination instantanée du défaut pour éviter que les parties saines de l'architecture, non concernées par le défaut, soient impactées par ses conséquences.

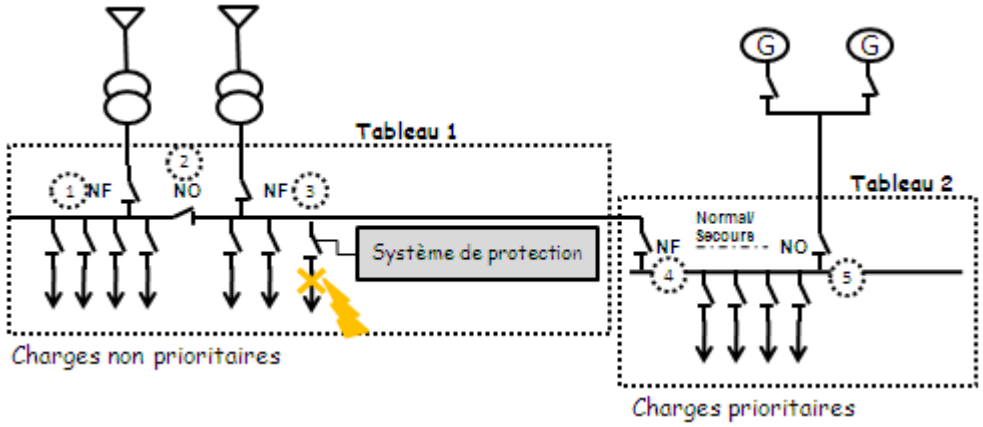


Figure 2.10 Emplacement d'un SDP dans une architecture électrique

A la différence des anciens SDP à base de relais électro mécaniques, les nouveaux SDP implémentés de nos jours dans un SDEE sont basés sur des microprocesseurs [Ozansoy et al, 2002]. La différence majeure entre les deux types de systèmes concerne le passage de l'analogique au numérique. Ce passage permet d'augmenter la capacité des nouveaux systèmes pour implémenter des nouvelles fonctionnalités de protection. Cet ajout de fonctionnalités permet de rendre les nouveaux SDP plus sophistiqués afin d'accroître le niveau de SDF des systèmes de distribution d'énergie. De plus, la numérisation des informations permet aux nouveaux SDP d'intégrer une fonction de communication permettant de raccorder le SDP à d'autres systèmes. Un échange d'informations pourra ainsi être établi entre les systèmes qui partagent le même réseau de communication. Nous allons voir au cours de ce chapitre que cet échange d'information est à l'origine des fonctionnalités d'automatisme avancées qui permettent la gestion du SDEE.

Par conséquent, comme le montre la figure 2.11, un SDP se composera de quatre unités fonctionnelles :

- une unité de mesure permettant l'acquisition des données électriques,
- une unité de contrôle permettant de lancer les procédures de protections en fonction des données reçues par l'unité de mesure,
- une unité de commande assurant le déclenchement et l'enclenchement des éléments de coupure en fonction des résultats de l'unité de contrôle et finalement
- une unité de communication permettant l'échange à distance des informations et des commandes entre le SDP et d'autres systèmes existants sur le même réseau de communication.

Le raccordement typique entre le SDP et l'élément de coupure est réalisé par l'intermédiaire des transformateurs de courant et de tension [Apostolov et al, 2010b]. Ces transformateurs

permettent d'adapter les grandeurs des mesures électriques pour les rendre accessibles par l'unité de mesure des SDP. L'unité de mesure permet la numérisation des valeurs analogique et leur envoi à l'unité de contrôle qui constitue la partie intelligente d'un SDP. Les fonctionnalités de protection sont développées sous la forme de programmes informatiques et implémentées dans cette unité. A l'issue des calculs réalisés, l'unité de contrôle envoie les informations calculées à l'unité de commande qui se chargera de déclencher ou enclencher les éléments de coupure.

Les unités composant le SDP peuvent être réparties dans divers équipements physiques ou elles peuvent être regroupées dans un seul équipement.

[Apostolov et al, 2004] présente une architecture composée de quatre équipements constituant un SDP. Les auteurs indiquent que dans une telle architecture, chaque unité est équipée d'une interface de communication lui permettant d'échanger des informations de mesure/contrôle/commande avec l'autre.

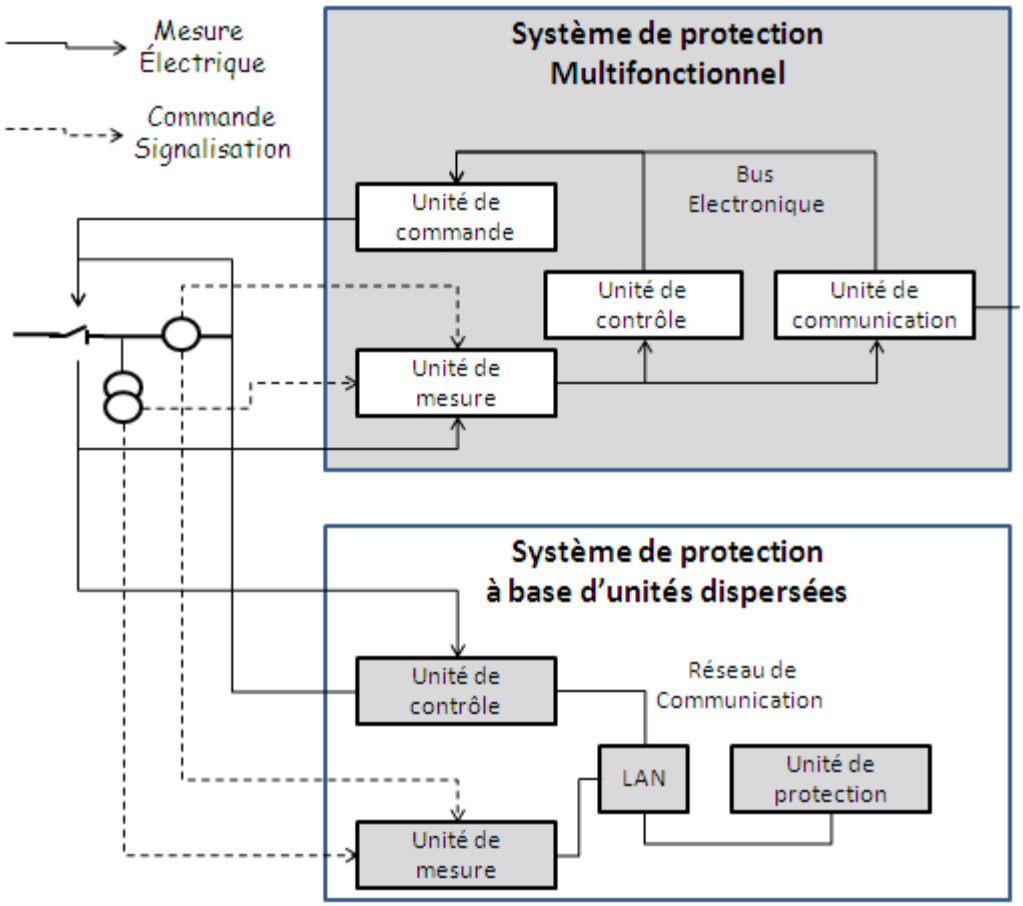


Figure 2.11 Diagramme fonctionnel des SDP

[Ren et al, 2010] spécifie que les SDP actuellement utilisés sont à la base un seul équipement embarquant toutes les unités essentielles décrites ci-dessous. Cet équipement est constitué d'un microprocesseur et est connu sous le nom d'équipement électronique intelligent multifonctionnel ou multifunctional IED. La connectique entre les différentes unités de cet équipement est faite soit en logique interne soit par des bus électroniques internes à l'équipement. Un tel SDP implémente une seule interface de communication associée à son unité de communication.

2.4.5 Les fonctionnalités principales de protection

Deux types de protection peuvent coexister dans un SDEE : les protections locales décrites dans ce paragraphe et les protections à base d'automatismes distribués décrites dans le paragraphe 2.4.6.

Les protections locales ont leurs fonctionnalités qui sont implémentées dans l'unité de contrôle du SDP.

(ANSI/IEEE C37.2-1979) établit des règles pour la numérotation des fonctionnalités locales disponibles dans les SDP du marché. Pour donner un exemple le code 50 symbolise la fonctionnalité de protection de la surintensité de courant électrique. Les détails de chacune de ces protections sont expliqués dans [Guide de protection].

Certaines protections locales sont autonomes tandis que d'autres sont paramétrables et nécessitent donc une configuration de la part de l'utilisateur (i.e. généralement des électriciens) avant leur mise en état d'opération. La configuration des protections paramétrables est régie généralement en respectant une courbe de caractéristiques. Comme montré sur la figure 2.12, la courbe de caractéristiques contient deux paramètres essentiels à configurer [Guide de protection]:

- le seuil : C'est la valeur limite d'une mesure électrique au delà de laquelle le SDP basculera du mode non opératoire vers le mode opératoire pour la prise d'une décision de contrôle.
- le retard : C'est la valeur indiquant le délai temporel impliqué à partir duquel le SDP est effectivement opérationnel.

En fonction de ces paramètres deux allures peuvent exister pour la construction de la courbe de caractéristique : une dans le cas de retard fixe et une autre pour le retard inverse.

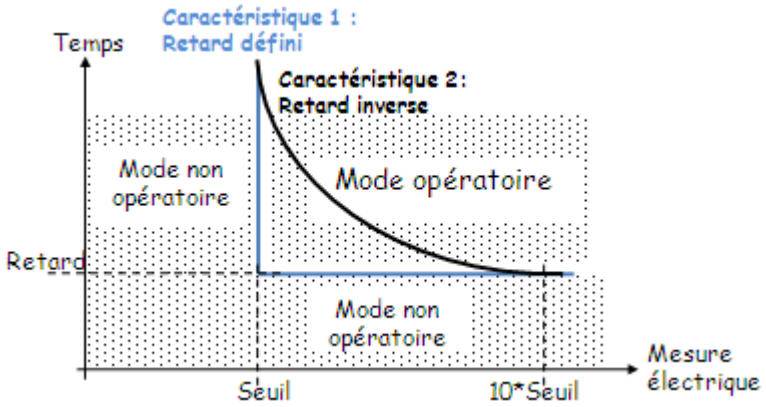


Figure 2.12 Courbe de caractéristique des protections locales paramétrables

La caractéristique 1 montrée sur la figure 2.12 consiste à déclencher la fonctionnalité après un retard fixe compté dès l'instant de dépassement du seuil. Pour la caractéristique 2, le retard de mise en mode opérationnel est d'autant plus faible que l'écart entre la mesure et le seuil est plus élevé.

2.4.6 Les applications d'automatismes distribués

Le fait que les fonctionnalités locales ne peuvent pas assurer complètement la protection d'un SDEE dans tous les cas de figure, entraîne l'obligation d'utiliser des protections à base d'automatismes distribués. Ces automatismes se basent sur une coordination entre les SDP et d'autres systèmes. Cette coordination assure une gestion technique efficace du SDEE.

En prenant comme exemple le tableau 2 de l'architecture de référence (figure 2.13) {chapitre 2, paragraphe 2.4.3}, et en supposant que l'un des groupes de production est tombé en panne, un scénario de délestage de charge non prioritaire devra être mis en œuvre pour respecter la condition que la puissance souscrite doit être égale à la puissance délivrée.

Les protections locales intégrées dans les SDP associés aux éléments de coupure ne sont pas capables d'effectuer en mode autonome cette procédure et ce du fait qu'ils n'ont pas une image de la mesure de puissance délivrée par les générateurs. Ainsi, un SDP coordinateur doit exister au niveau des jeux de barres. Ce système aura pour rôle de superviser la puissance et d'envoyer des ordres vers les SDP en aval pouvant aboutir en cas de besoin à un délestage de charges.

Ainsi, l'envoi des ordres nécessite la présence d'une fonctionnalité d'échange d'informations entre les différents SDP. Comme cela sera montré dans ce chapitre, différentes méthodes peuvent exister pour assurer cette fonctionnalité. Cet échange d'information sera ainsi à l'origine des protections à base d'automatismes distribués avancées.

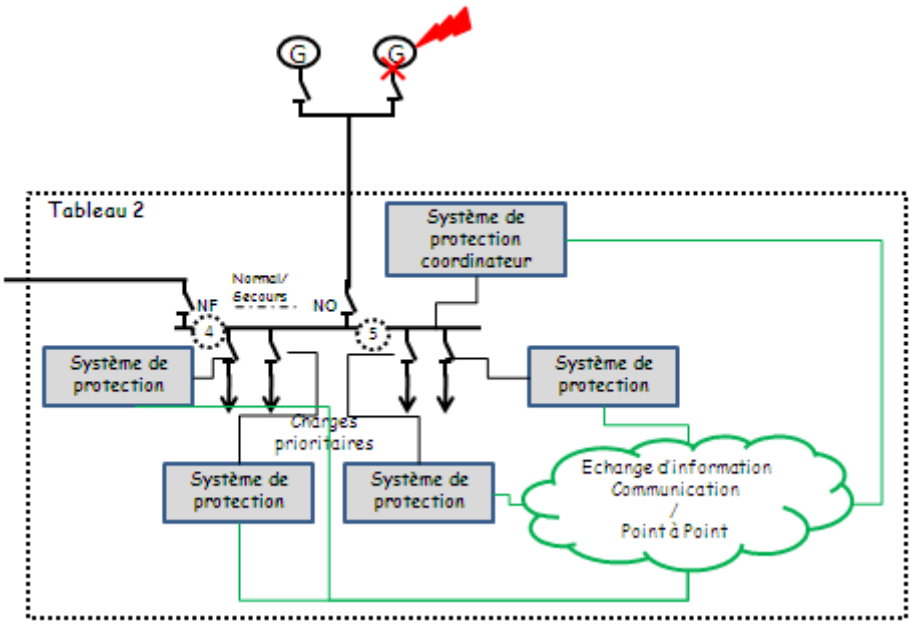


Figure 2.13 Coordination entre les SDP

2.5 Automatisme distribué pour des fonctionnalités de protection critiques

En général les systèmes d'automatisation sont définis comme étant composés d'instruments [Bayard, 1994], capteurs [Robert, 1992] ou actionneurs intégrant des unités de traitements à base de microprocesseurs. Lorsque ces systèmes intègrent une interface de communication, ils sont aptes à se connecter à un réseau de communication pour échanger des informations les uns avec les autres. Parmi les informations échangées, il pourra y avoir des échanges destinés à réaliser des missions critiques d'automatismes et dans ce cas on parlera d'automatismes distribués [Chotin et al, 1994].

Ce paragraphe va montrer les différents types de protections à base d'automatismes distribués entre les SDP implémentés dans un système global de distribution d'énergie. Certaines fonctionnalités listées dans ce paragraphe peuvent avoir des extensions ou peuvent ne pas être implémentées dans un SDEE comme c'est expliqué dans la spécification fonctionnelle réalisée par un groupe de travail espagnol regroupant les sociétés d'électricité essentielles en Espagne [E3 Group of Spanish Electricity companies for studies on IEC 61850, 2010].

L'objectif essentiel de ce paragraphe est d'expliquer les applications d'automatismes distribués les plus répandues dans l'industrie, et de montrer les différentes méthodologies, en termes d'échanges d'informations, permettant leur implémentation.

Pour terminer, une analyse de l'influence de ces protections distribuées sur les attributs de la SDF d'un SDEE est proposée.

2.5.1 La sélectivité

Lors de l'occurrence d'un défaut électrique, plusieurs SDP localisés dans différentes zones de l'installation peuvent le détecter. Par conséquent, une mise hors service de certaines parties de l'installation non directement concernées par le défaut peut se produire.

Pour cela une méthode doit être mise en place permettant de rendre le système bien sélectif de façon à ce que seul le SDP se trouvant en amont d'un défaut l'éliminera (i.e. faute interne à un composant de protection). Cette méthode est appelé la sélectivité [Hutchinson et al, 1996] (appellation anglophone **'Interlocking'**).

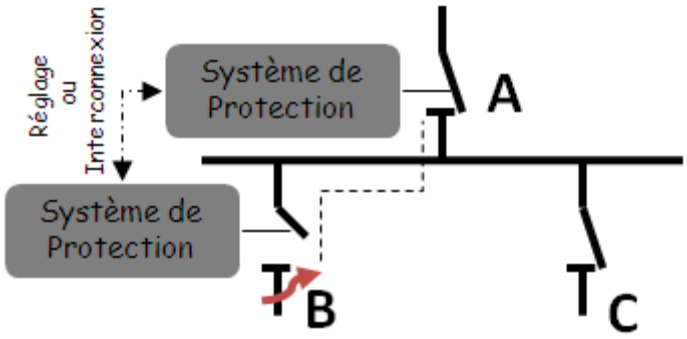


Figure 2.14: Principe de la sélectivité

La figure 2.14 illustre le principe de la sélectivité, la flèche située au point B associée au trait pointillé représente un défaut électrique qui remonte d'un étage aval vers un étage amont.

Le SDP situé au point B envoie une commande tout ou rien dans le but de déclencher le contact de l'élément de coupure associé. La sélectivité consiste à ce que le défaut détecté au niveau de B ne soit en aucun cas remonté à l'élément de coupure situé en A, pour garder la branche C sous alimentation électrique.

Pour assurer cette fonctionnalité d'automatisme, les SDP doivent se mettre dans un état de coordination pour indiquer que le défaut électrique est bien pris en compte par l'un des SDP et devra être ignoré par l'autre (i.e. dans l'exemple le SDP en B est celui qui prend en charge l'élimination du défaut et le système en A devra ainsi l'ignorer).

Plusieurs moyens existent pour assurer cette coordination :

- réglage des SDP

- interconnexion physique entre les SDP pour échanger les informations d'automatismes

Chacun de ces moyens pourra être réalisé en utilisant différentes techniques [Sautorio, 1990]. Les techniques ampérométrique et chronométrique se font par l'intermédiaire du réglage des SDP tandis que les techniques logiques et de communication se basent sur des interconnexions entre les SDPs.

2.5.1.1 Technique ampérométrique

La sélectivité ampérométrique [Sautorio, 1990] se base sur le principe de réglage des seuils de SDP. Cette technique s'appuie sur l'hypothèse que l'intensité de court-circuit est d'autant plus élevée que le défaut est proche de la source de production.

En prenant l'exemple de la figure 2.14 le seuil du SDP situé en A doit être configuré avec une valeur plus élevée que celui situé en B. La correspondance de ce type de réglage avec les courbes de caractéristique des deux SDP est illustrée sur la figure 2.15.

Toutefois, cette technique présente des limites d'utilisation et ne fonctionne pas pour n'importe quel point de fonctionnement. En effet en considérant un défaut au point B excédant la valeur du seuil du SDP en A, il sera éliminé en même temps par les deux SDP. Donc, la sélectivité utilisant cette technique est considérée comme étant une sélectivité partielle.

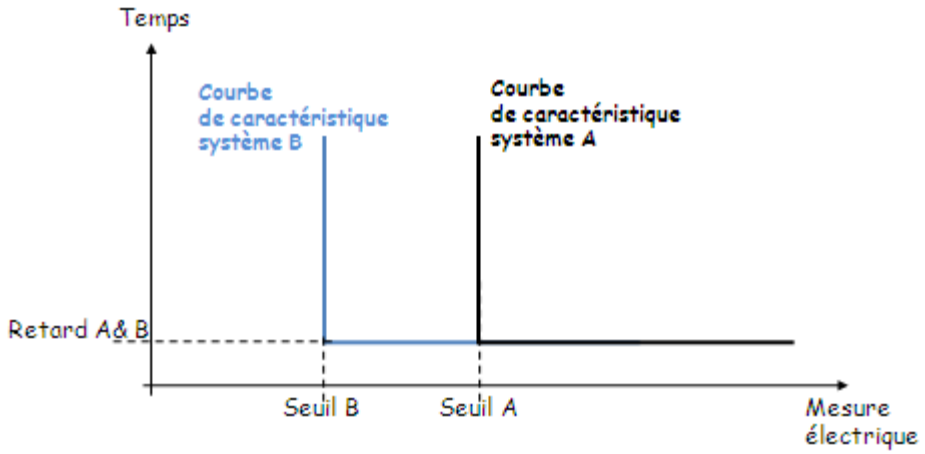


Figure 2.15 Sélectivité ampérométrique

2.5.1.2 Technique chronométrique

La sélectivité chronométrique [Guide de protection] se base sur le réglage des retards des SDP de façon à être d'autant plus élevé que le SDP est plus proche de la source (i.e. le plus en amont dans l'architecture). Par conséquent, comme le montre la figure 2.16, le réglage du retard d'un SDP amont doit être plus élevé que celui du SDP aval.

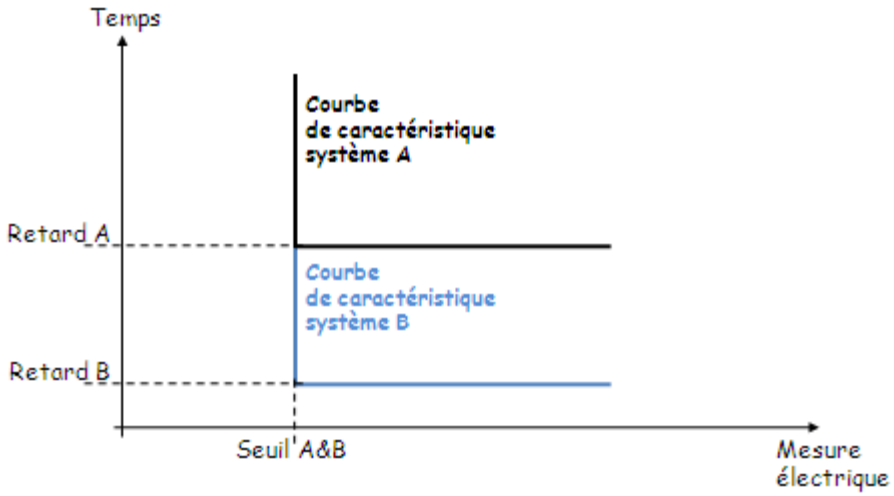


Figure 2.16 Allure des courbes de caractéristique des SDP aval et amont pour une sélectivité ampérométrique

Pour le bon fonctionnement de cette technique, la différence entre les retards de deux SDP amont et aval doit être plus grande que le temps d'isolement de l'élément de coupure aval additionné du temps de désexcitation de l'élément de coupure en amont (figure 2.17).

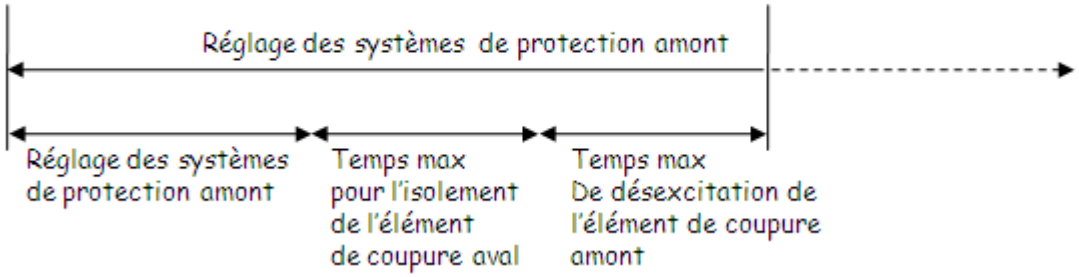


Figure 2.17 Réglages du SDP pour la technique chronométrique

L'avantage de cette technique est qu'elle assure sa propre redondance ; en effet en supposant que le défaut n'ait pas été éliminé au point B pour une raison quelconque, il sera repris en compte et éliminé au point A. Néanmoins, cette technique présente des conséquences néfastes dues à l'accumulation des retards de temporisation. En effet, ce retard peut atteindre la valeur 0,9 seconde à la source ce qui impose une incompatibilité avec les impératifs du fournisseur d'énergie demandant une temporisation plus courte au niveau du poste de livraison. D'autre part, l'utilisation de cette technique exige un surdimensionnement thermique des câbles et des tableaux pour pouvoir supporter les délais de défauts dus à sa temporisation.

2.5.1.3 Technique logique

La technique logique est basée sur le principe d'interconnexion entre les SDP par l'intermédiaire de liaisons filaires point à point. La schématique de connexion de base

consiste à relier les sorties des SDP aval aux entrées des systèmes amonts par l'intermédiaire de cette interconnexion filaire.

Cette technique est brevetée par Merlin Gerin [brevet n° 1 421 236] et s'est imposée aux électriciens jusqu'aux années 2000. Son principe se base sur un envoi des informations tout ou rien pour le blocage des systèmes en amont lors de la prise en compte du défaut par ceux en aval. Ainsi comme l'indique la figure 2.18, à l'apparition d'un défaut au point B, son élimination sera retardée en respectant le paramètre de retard du SDP en B. En même temps une commande instantanée est envoyée d'une des sorties du SDP en B vers une entrée du SDP en A. A la réception de cette commande, le SDP en A inhibera la procédure de la commande retardée de l'élimination du défaut en question.

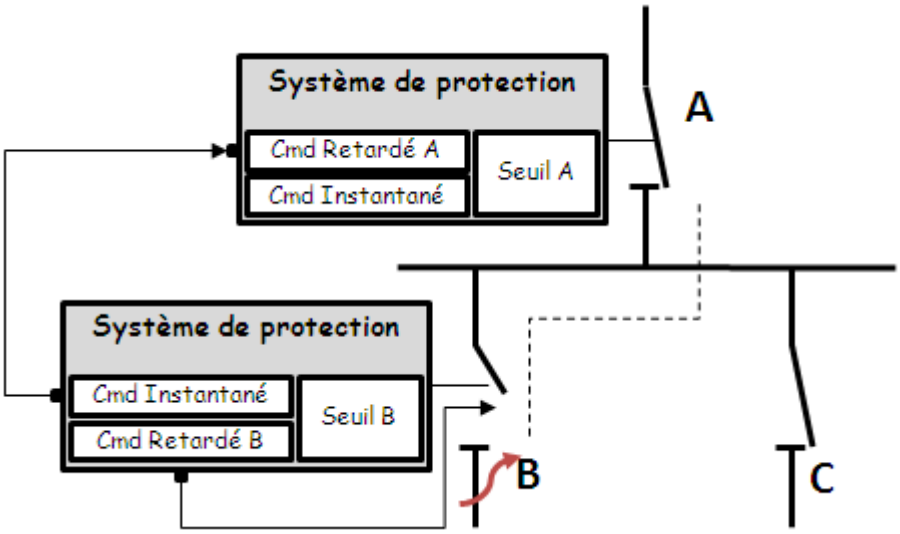


Figure 2.18 Schéma d'interconnexion pour la technique de la sélectivité logique

Cette technique permet d'obtenir une sélectivité au déclenchement totale, mais elle présente des inconvénients au niveau des coûts d'installations à cause des longues interconnexions câblées entre les SDP [Guide de protection] (e.g. elles peuvent atteindre parfois plusieurs centaines de mètres).

2.5.1.4 Technique de communication

[Ali et al, 2008] [Thomas et al, 2010] proposent des systèmes de distribution électrique intégrant un réseau de communication. Ce réseau est configuré de manière à pouvoir communiquer des messages de communication critiques entre les SDP; ces messages sont bien sûr temps réel. Dans leurs publications, les auteurs expliquent que les messages de communication temps réel sont essentiellement dédiés à la réalisation d'applications de protections distribuées. La sélectivité fait partie d'une des applications d'automatismes cités par les auteurs. Cette technique, intitulée technique de communication, est considérée comme l'une des technique la plus récente dans le monde des automatismes distribuées dans un SDEE. En utilisant cette technique, le réseau de communication sera non seulement un moyen pour connecter les SDP au système de supervision et contrôle commande, mais aussi un média permettant l'interconnexion entre les SDP pour l'échange des messages critiques d'automatisation.

Des trames de communication transiteront entre les interfaces réseaux des SDP pour remplacer les entrées/sorties de la technique logique.

Les avantages d'un tel système par rapport à un système traditionnels basé sur des connexions câblés point à point sont nombreux :

- réduction de câblage
- augmentation de flexibilité et de re-configurabilité
- potentiel de diagnostic et supervision des informations depuis une supervision

Néanmoins, l'utilisation d'une telle technique nécessite une étude précise de la disponibilité du réseau de communication et de la fiabilité en termes d'étude de performance des messages critiques dédiés aux applications d'automatisme distribués. Pour cela des architectures à base de commutateurs réseaux avec/sans notions de priorité (IEEE 802.1p), et avec/sans notion de VLAN (IEEE 802/1q) sont proposés par les auteurs dans le but de déterminer des stratégies plus adaptées aux exigences temporelles des applications d'automatisme distribuées.

2.5.2 Automatisme de vote

Le concept de cet automatisme est publié dans [IEEE PSRC H6 : Special Report, 2005]. Son principe consiste à ce que lors de l'apparition d'un défaut sur un bus électrique protégé par plusieurs SDP, ces derniers doivent échanger entre eux des messages confirmant que le défaut est détecté par plus d'un système. Ainsi, la procédure de commande d'un SDP sera associée à une logique combinatoire illustrée sur la figure 2.19.

L'échange d'information entre les SDP est établi avant la prise d'une décision d'isolement du bus électrique. Selon les auteurs de [IEEE PSRC H6 : Special Report, 2005], cet échange d'information se base sur l'interconnexion entre les SDP par l'intermédiaire d'un réseau de communication.

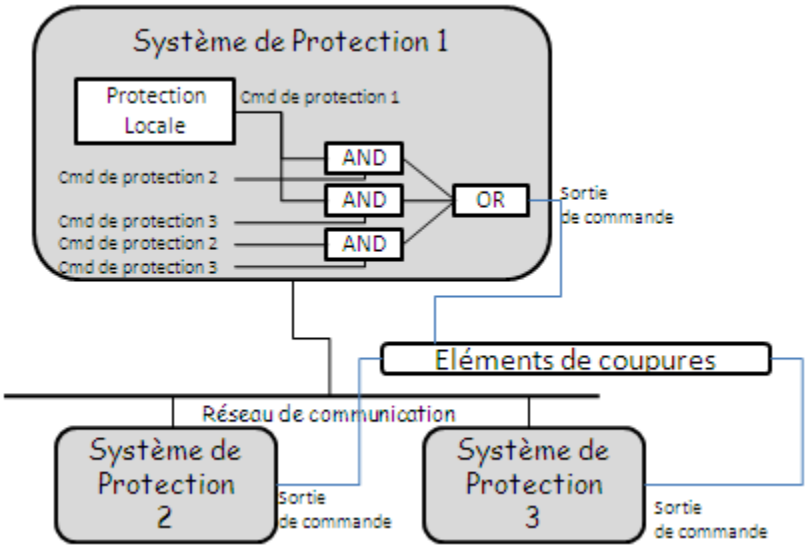


Figure 2.19: Principe de l'automatisme de vote

2.5.3 Automatisme de protection de défaillance des éléments de coupure

Ce paragraphe illustre le concept de l'automatisme distribué entre les SDP dans le cas de l'occurrence d'une défaillance au niveau d'un élément de coupure. Le détail de ce principe est expliqué dans [Guide of the IEEE Power society].

La défaillance des éléments de coupure est définie comme étant l'incapacité de ces éléments à ouvrir leurs contacts électriques suite à la réception d'une commande d'ouverture issue du SDP. Ainsi l'interaction entre les éléments de coupures et les SDP sera interrompue.

Plusieurs causes peuvent être à l'origine de cette défaillance:

- mécanique de l'élément de coupure
- défaut électronique au niveau du contact de l'information envoyée par le SDP
- défaut d'alimentation électrique des éléments de coupures

En fonction des exigences imposées à certaines parties du SDEE, la défaillance d'un élément de coupure pourra engendrer l'appel à d'autres éléments de coupure pour isoler les sources qui ont causé l'apparition du défaut électrique.

Le principe de base de cet automatisme est réalisé par l'intermédiaire d'une logique intégrée dans les SDP à base de microprocesseur.

Cette logique consiste à déclencher un temporisateur à l'issue de l'apparition de 2 signaux :

- le premier signal indique la détection d'une décision de commande d'ouverture
- le deuxième signal est une information détectée par un élément de mesure de courant électrique indiquant un passage de courant au travers de l'élément de coupure

La valeur du délai de temporisation est composée des paramètres suivants :

- le temps requis par l'élément de coupure pour éliminer le défaut
- le temps correspondant à la désexcitation de l'élément de détection de courant
- une marge de temps supplémentaire de sécurité

L'expiration du temps de temporisation indique une défaillance au niveau de l'élément de coupure. Cette expiration sera traduite par le déclenchement d'un contact qui sera à l'origine du transfert de l'information de défaillance aux SDP redondants.

Si ces systèmes se trouvent en local ou sont distants par rapport à l'élément de coupure défaillant, l'échange de l'information entre les systèmes peut être assuré par des moyens différents.

En prenant le cas d'un SDP local à l'élément de coupure défaillant, le médium d'échange des informations qui est le plus souvent utilisé dans ce cas est l'échange filaire point à point. Tandis que pour les SDP distants, le réseau de communication sera le médium utilisé pour l'échange de l'information.

Le diagramme montré sur la figure 2.20 illustre les différents paramètres temporels qui entrent en jeux dans un cet automatisme. Le rectangle montré sur cette figure symbolise le retard de la transmission de l'information du SDP de l'élément défaillant vers le SDP distant au travers d'un réseau de communication.

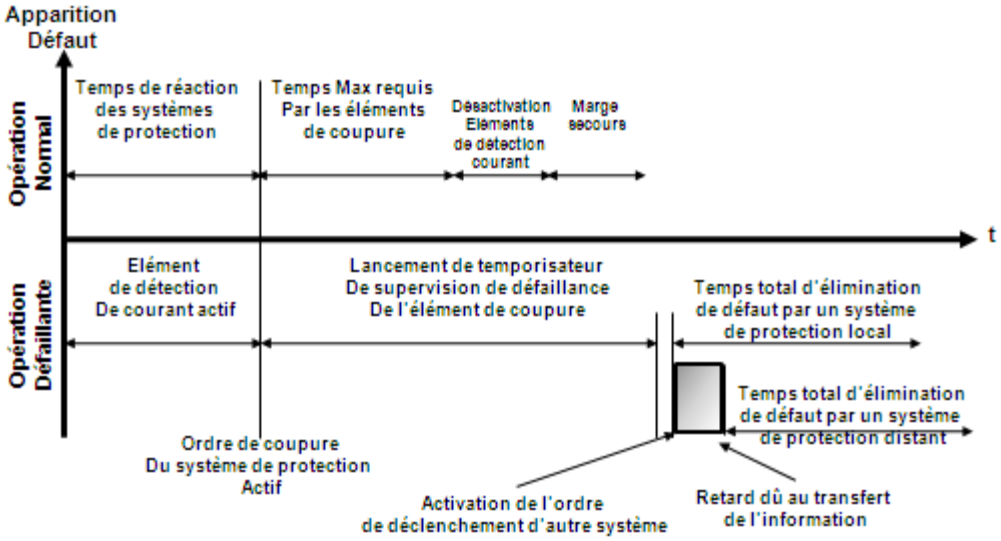


Figure 2.20: diagramme temporel de l'automatisme de défaillance

2.5.4 Automatisme de délestage et relestage

Une gestion de la puissance électrique doit être mise en place dans un SDEE. Le principe se base sur le fait que la puissance souscrite ne doit pas dans la mesure du possible excéder la puissance fournie par le poste de livraison. Le non respect de cette condition contribue à des coûts de pénalités de dépassement dans le cas d'un fournisseur d'électricité externe ou à un écroulement des générateurs dans le cas d'une production d'énergie interne.

Pour assurer une telle gestion des procédures d'automatisme doivent être mise en place :

- mise en marche des groupes de production interne aux sites
- délestage des consommateurs selon des niveaux de priorités. Pour une gestion avec n niveaux de priorités, le délestage s'effectuera en commençant par le niveau n puis sera progressif si nécessaire.

Plusieurs approches, décrites dans les sous-paragraphes suivant, existent pour accomplir la mission de délestage [Shokooh et al, 2005].

2.5.4.1 Approche de coordination des éléments de coupure

Cette approche est la méthode la plus simple à utiliser pour l'implémentation des automatismes de délestage/relestage. Elle consiste à relier par l'intermédiaire **des liaisons filaires** l'élément de coupure d'une source d'alimentation à l'ensemble des éléments de coupures des récepteurs. Le principe se base sur le fait que lors de la perte d'une source d'alimentation, un signal tout ou rien sera directement envoyé de son élément de coupure vers les récepteurs dans le but de les délester.

Cette approche présente des avantages au niveau temporel du fait qu'il n'y a aucune intelligence intégrée et que les signaux de commande au déclenchement sont directement envoyés aux récepteurs.

Néanmoins les points listés ci-dessous constituent des inconvénients majeurs de cette approche :

- un seul niveau de délestage des récepteurs est présent,
- une charge maximale est délestée correspondant à un calcul pire cas quelle que soit la nature de la discordance qui a eu lieu,
- la modification de l'architecture de délestage est coûteuse et permet dans certains cas un arrêt d'une partie de l'installation.

2.5.4.2 Approche du SDP conventionnel

Cette approche consiste en la mise en place d'un SDP conventionnel, numéroté (81) (ANSI/IEEE C37.2-1979), dans le but de gérer le principe de délestage. Ce système permet la supervision de la fréquence du réseau électrique et l'envoi des commandes de déclenchement aux autres SDP en fonction de la table de fréquence configurée.

Par exemple, un pourcentage de 10% de réduction de charge pourra être configuré pour une perte de fréquence de 0,5%. Si le système (81) détecte toujours une baisse de fréquence, il déclenchera une autre plage de charge lorsqu'il atteint un nouveau seuil de fréquence configuré.

Un des inconvénients lié à cette approche est le fait que chaque seuil introduira un retard au délestage du seuil suivant. Ceci pourra ainsi causer une instabilité électrique due à une surcharge des groupes de production et pourra causer dans certaines circonstances l'écroulement des générateurs avant le délestage des récepteurs.

Un autre inconvénient lié à cette approche est qu'à l'issue d'une détection d'une baisse de fréquence, le SDP (81) envoie les commandes de déclenchement aux éléments de coupure configuré au préalable dans ce système sans la moindre connaissance de la puissance consommée instantanée.

On peut conclure de ces inconvénients que cette approche n'est pas optimale pour la réalisation du principe de délestage.

2.5.4.3 Approche par automates programmables

Cette approche consiste à utiliser les automates programmables industriels comme équipements intelligents pour l'approche de délestage/relestage.

Généralement l'intelligence implémentée dans l'automate programmable est réalisée de manière à lancer la séquence de délestage des récepteurs après avoir fait le calcul du bilan des puissances livrées et consommées. Un calcul de fréquence pourra de même être fait dans l'automate pour remplacer le calcul de puissance.

L'un des avantages majeurs de cette approche est le fait que la modification de la séquence de délestage nécessite un simple changement au niveau du programme d'automate.

Une architecture typique se basant sur ce principe est réalisée par une équipe d'ingénierie d'Euro System. Cette architecture est divisée en une centrale électrique et plusieurs postes de transformation. Dans la centrale électrique les automates programmables sont connectés aux SDP des groupes de production internes. Cette connexion est établie par l'intermédiaire **d'un réseau de communication Ethernet**. Cette connexion permet de transmettre les informations relatives aux puissances calculées dans les SDP vers l'automate programmable. En fonction

de ces informations des automatismes de délestage programmés dans l'automate seront lancés de la manière suivante :

- Lors d'une perte de l'alimentation EDF, l'automate envoie des ordres de délestage à toutes les charges délestables et met en route les trois groupes de production d'énergie interne. La mise en marche des groupes entraîne l'appel à la procédure de relestage des charges en allant de la priorité la plus élevée vers la priorité la plus basse. La notion de priorité est implémentée dans une table de mots interne à l'automate.
- Lors de la perte d'un des groupes de production, l'automate recevra une information des SDP indiquant qu'il y a une charge supplémentaire au niveau de l'installation. La séquence de délestage sera ainsi appelée jusqu'à ce que l'information d'excès de charge disparaisse.

[Saxena et al, 2010] indique qu'en utilisant cette approche, l'aspect de changement des niveaux de priorités de délestage pourra se faire par une simple mise à jour de cette table des priorités par un système de supervision-contrôle-commande. Ceci contribue à une meilleure gestion au niveau du process de l'installation et permet de suivre l'évolution de l'environnement.

La figure 2.21 montre qu'à la différence de l'approche conventionnelle, une seule étape de délestage est requise avec une approche à base d'automates programmable. La réduction des étapes contribue à la minimisation des perturbations sur le réseau électrique. L'approche à base d'automates consiste à prendre la bonne décision de délestage dès la première tentative en passant de la puissance nominale vers une nouvelle valeur de puissance, intitulée puissance d'équilibre et indiquée **'Peq'** sur la figure, après le délestage des consommateurs. Néanmoins l'approche conventionnelle doit quant à elle passer par plusieurs étapes de délestage pour atteindre la puissance d'équilibre souhaitable.

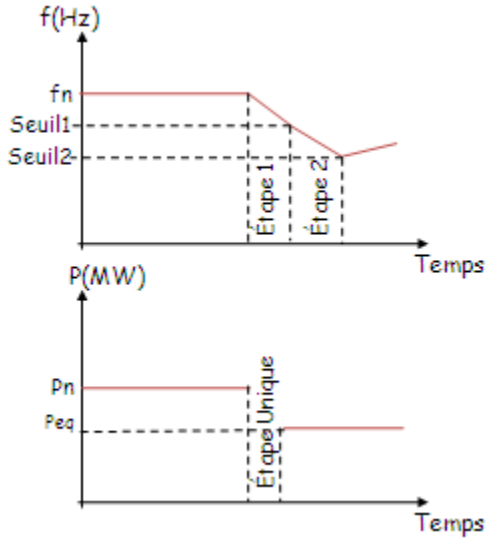


Figure 2.21: Comparaison entre l'approche conventionnelle et l'approche d'automates

2.6 Influence des automatismes distribués sur la SDF d'un SDEE

Ce paragraphe va montrer l'influence des protections à base d'automatismes distribués sur les attributs de la SDF d'un SDEE.

L'absence du principe de délestage/relestage peut contribuer à une charge excessive sur le réseau électrique en risquant de le mettre dans un état de perturbation. Ces perturbations peuvent causer la perte du groupe de production d'énergie qui peut se traduire par un arrêt complet du système pendant la durée de sa mission. Ainsi, en faisant la correspondance avec les attributs de la SDF, on pourra conclure que cet automatisme aura un effet direct sur la fiabilité du SDEE.

La protection d'un SDEE est considérée efficace si elle peut traiter tous les cas d'anomalies pouvant apparaître sur le réseau électrique. Dans ce cas, un défaut électrique pourra être détecté et contrôlé par plusieurs SDP appartenant au SDEE. En prenant le cas où le défaut est traité par un SDP en aval et un autre en amont, l'absence d'un système sélectif affectera l'arrêt de certaines branches saines du SDEE. Ainsi, l'automatisme de sélectivité a une influence directe sur l'attribut de la disponibilité du SDEE. En revanche, dans certains cas où le SDP aval est dans l'incapacité à réagir, l'élément de coupure se trouvant en amont pourra être vu comme une redondance. Donc même si la sélectivité permet d'augmenter le niveau de disponibilité du système sauf si elle n'est pas bien étudiée, elle pourra chuter la valeur de la fiabilité du système dans le cas où la redondance est un paramètre important.

A son tour l'automatisme lié à la défaillance des éléments de coupure affecte la sécurité du SDEE. En effet à l'apparition d'une anomalie électrique, dans une sous-station ou une branche de l'installation, si elle n'est pas instantanément éliminée par l'élément de coupure juste en amont, elle pourra engendrer des instabilités au niveau électrique dans la partie correspondante à la défaillance ; ceci peut introduire parfois la destruction des récepteurs.

Finalement, l'automatisme de vote pourra être vu comme un système permettant d'améliorer la fiabilité du SDEE. En effet, le fait d'éviter la mise hors service de certaines parties de l'installation suite à une décision fautive du SDP permet d'améliorer la fiabilité du système en question. En revanche, l'augmentation de la fiabilité en question peut contribuer à mettre en jeu l'attribut de sécurité : en effet un retard pour une prise de décision de déclenchement induit généralement des perturbations sur le réseau électrique.

2.7 Conclusion

Chaque système doit être muni d'une étude détaillée de sa SDF. Une explication détaillée des caractéristiques de la SDF a été abordée dans ce chapitre. Les moyens de la SDF sont ceux qui permettent d'évaluer les pertinences des attributs de cette dernière.

Un SDEE est système global contenant plusieurs autres systèmes imbriqués. Les SDP font partie des sous-systèmes les plus essentiels de ce système global.

Une partie de cette protection peut être réalisée en local aux SDP, tandis que l'autre nécessite un échange d'information entre plusieurs SDP. Cette dernière partie est intitulée protection à base d'automatismes distribués. Ces applications sont indispensables à mettre en place dans les SDEE du fait qu'elles contribuent à une amélioration nette des attributs de SDF de ce dernier.

L'absence de ces automatismes peut causer un black-out complet du SDEE. En prenant le cas d'une absence d'un système sélectif, un défaut électrique peut atteindre la source de production en remontant les différents niveaux de hiérarchie électrique. Par ailleurs, en absence de la technique de délestage de charge, un écroulement des groupes de production peut avoir lieu à cause de la surcharge du réseau électrique.

L'échange d'informations entre les SDP est basé soit sur un média filaire point à point soit sur un réseau de communication. Dans le dernier cas, il sera indispensable de bien étudier et évaluer les performances des messages de communication en les comparant aux exigences temporelles imposées par les applications d'automatismes distribuées montrées dans ce chapitre.

Plusieurs approches permettant l'évaluation de performances des applications distribuées vont être abordées dans le chapitre 3. Un choix particulier d'une approche spécifique est discuté également dans le chapitre suivant.

Chapitre 3

Standard de communication IEC61850

3.1 Introduction

Ce chapitre aborde les caractéristiques d'un nouveau standard de communication, intitulé 'IEC 61850', dédié aux équipements de communication des architectures de distribution d'énergie électrique. Ce standard est diffusé mondialement pour permettre une utilisation interopérable entre les différents constructeurs des systèmes de protection.

Le paragraphe 2 aborde l'importance de ce standard par rapport aux protocoles de communication conventionnels utilisés dans les architectures de distributions électrique.

Le paragraphe 3 poursuit pour expliquer l'interopérabilité pourvue par le biais d'une implémentation d'un dictionnaire standard de données électriques décrites sous format objets compatibles aux SDEE.

Les services ainsi que les différents profils de communication, conventionnels et avancés, sont abordés dans le paragraphe 4.

Le paragraphe 5 explique la manière de configuration d'un système de communication basé sur le standard IEC 61850.

Finalement, le paragraphe 6 décrit l'une des fonctionnalités les plus importantes du standard qui permet l'échange des signaux rapides 'temps réel' au niveau du réseau de communication. Cette fonctionnalité existe dans le but d'assurer un moyen d'échange d'information critique entre les systèmes de protection pour leur permettre de réaliser les protections distribués, qui étaient traditionnellement accomplis par des connexions filaires point à point [Hakala-Ranta et al, 2009]. Le principe de fonctionnement ainsi que les différentes applications pouvant être réalisées par cette fonctionnalités sont abordés dans le paragraphe.

3.2 IEC61850 : nouveau standard de communication dans un SDEE

Dès le début des années 1990, les technologies de communication sont apparues dans les systèmes de distribution d'énergie. Différents protocoles de communication ont été définis par les constructeurs et implémentés dans les unités de protection, dans le but de définir des règles permettant à ces unités d'échanger leurs informations avec d'autres composants du système global.

Par l'utilisation de ces protocoles, un accès/une connexion entre les différents étages (i.e. SDP, système de contrôle commande) d'une installation pourrait s'établir.

L'ISO a divisé le traitement de la communication en plusieurs couches : physique, liaison, réseau, transport, session, présentation et application. Ces couches sont connues par le modèle OSI. Elles permettent de définir la manière dont les données transitent d'un composant à un autre dans un même réseau de communication. Chacune de ces couches implémente un protocole de communication bien défini. Pour établir une communication entre deux équipements réseau, les protocoles de communication existants dans les couches des deux composants doivent être identiques [Ozansoy et al, 2002].

Chaque constructeur étant sur le marché a imposé son propre protocole de communication au niveau de la couche application. [Adamiak et al, 1998] fait état des différents protocoles

applicatifs utilisés dans une architecture de communication d'un SDEE. Selon les auteurs, le ModBus est le protocole de communication le plus répandu dans de telles architectures. Ce protocole fait l'objet de nos premiers développements de modélisation expliqués dans les chapitres 5 et 6.

La variété de ces protocoles de communication au niveau de la couche applicative engendre des impacts négatifs sur les architectures de communication. Nous allons lister ci-dessous les impacts les plus sensibles du point de vue de l'utilisateur final :

- limitation du choix des composants constitutifs du SDEE
- introduction d'équipements additionnels à l'architecture tels que les convertisseurs de protocole et les concentrateurs de données [Haffar et al, 2009]
- dégradation des performances du réseau de communication à cause des équipements tiers rajoutés

Pour toutes ces raisons, la **C**ommission **E**lectrotechnique **I**nternationale **CEI** (ou **I**nternational **E**lectrotechnical **C**omission IEC) a proposé une standardisation du protocole de communication dans un SDEE.

Les caractéristiques de ce standard constituent l'épicentre de ce travail de recherche. Ces caractéristiques sont expliquées dans le paragraphe suivant.

3.3 Interopérabilité et dictionnaire d'objets du standard IEC 61850

La standardisation d'un protocole de communication consiste à définir un ensemble de données et d'échanges de communication standard, au domaine d'application utilisant ce protocole.

Cette définition est basée sur l'implémentation d'un dictionnaire associé au domaine d'application du langage de communication en question. Ce dictionnaire doit être diffusé et utilisé mondialement par la majorité des constructeurs afin que le protocole soit perçu comme un standard.

N'importe quel langage est basé sur deux termes essentiels constituant le noyau de sa construction: les noms et les verbes [Adamiak et al].

Les noms constituent des ensembles de données regroupés dans une structure ayant une nomenclature bien définie [Adamiak et al]. Cette structure est intitulée **'Objet'** du point de vue de la norme IEC 61850. Les objets peuvent s'encastrer entre eux pour produire des structures de données plus complexes. L'objet élémentaire est celui qui décrit la variable physique d'un équipement. Le plus grand objet d'un équipement est celui qui donne une vue globale de toutes les variables physiques ou logiques de l'équipement.

Les verbes sont considérés, en quelque sorte, comme les services de communication échangés entre les équipements. Ces services permettent un accès aux objets (i.e. variable physique ou logique) implémentés dans les équipements. La condition nécessaire pour réussir cet accès consiste à ce que les équipements se trouvent sur un même réseau de communication.

La première fois d'un tel principe de communication est apparu remonte aux années 1980 dans le domaine de la **G**estion **T**echnique des **B**âtiments **'GTB'**. Le standard de communication interopérable était le **'LonWorks'** [BTIB]. Ce protocole a connu beaucoup de

succès et est actuellement installé dans les architectures de communication des bâtiments du monde entier (e.g. France Infogrammes [Le Men Newron System]).

La standardisation des protocoles de communication dans les architectures de communication d'un SDEE apparut pour la première fois en 1994 et ce, avec le premier standard **U**tility **C**ommunications **A**rchitecture **UCA,** conçu par **E**lectric **P**ower **R**esearch Institute, Inc. **'EPRI'**. Ce standard a été publié par **I**nstitute of **E**lectrical and **E**lectronics **E**ngineers **'IEEE'** en 1999 [IEEE-SA 1999].

Un comité technique formé de 57 experts dans le domaine électrique **TC57** a commencé à travailler sur la norme IEC61850 en 1996, dans un but similaire à celui de l'UCA, pour définir un standard de communication pour le monde de la distribution d'énergie [INTERNATIONAL STANDARD IEC 61850-1].

Ces 2 groupes se sont réunis en 1997 pour unifier leur développement et définir un standard commun réunissant les points essentiels définis par chacun. Leur collaboration a contribué à la publication, en 2003, de la première édition du nouveau standard de communication IEC61850 [Hoga et al, 2005a] [INTERNATIONAL STANDARD IEC 61850-1].

La notion d'interopérabilité constitue l'aspect le plus important du protocole IEC 61850. Ce protocole peut être considéré comme un standard du fait qu'il est retenu par tous les grands constructeurs du monde [Chen et al, 2008].

Cette notion est définie par plusieurs articles [Hoga et al, 2005b] comme étant l'aptitude d'un équipement à opérer avec d'autres équipements sur le même réseau de communication. Cette coopération a pour but d'assurer le partage de fonctionnalité entre les équipements, par l'intermédiaire de l'échange des informations et des commandes.

Du point de vue des utilisateurs, cette notion est confondue par l'interchangeabilité définie comme étant l'aptitude d'un équipement à être remplacé par d'autres équipements qui ne proviennent pas d'un même constructeur [Janssen et al, Kema T&D Power].

Une condition nécessaire, non suffisante, pour garantir l'interopérabilité d'une architecture de communication, consiste à avoir une certification d'implémentation du standard IEC 61850 pour tous les équipements constituant cette architecture [Tan et al, 2008].

3.3.1 Formats des objets IEC 61850

Le standard IEC 61850 impose un format d'objet hiérarchique qui doit être implémenté dans les équipements certifiés conformes à cette norme [Apostolov et al, 2003]. La figure 3.1 illustre le principe de composition de ces objets.

Les unités de protection sont les équipements essentiels destinés à adapter ce standard de communication. La partie serveur de ce standard de communication est celle qui sera implémentée dans ces unités. Au niveau de la communication, ces unités seront perçues comme des serveurs IEC 61850.

Suite à cette implémentation, un serveur aura une vue logique symbolisée de conteneur d'objets. Le serveur sera intitulé **P**hysical **D**evice **'PD'** au sens de la norme. Un **'PD'** est réparti de manière hiérarchique sous forme de **L**ogical **D**evice **'LD'**, **L**ogical **N**ode **'LN'**, **D**ata **O**bject **'DO'** et **D**ata **A**ttributes **'DA'**.

Traditionnellement, un **'PD'** est composé d'un seul **'LD'** décrivant son contenu. Néanmoins, cette hypothèse ne peut pas s'appliquer à tous les équipements existants contenus dans un

SDEE. Prenons l'exemple d'un concentrateur de données : il est conçu pour se connecter à plusieurs unités de protection afin de concentrer les données essentielles venant de ces unités et de les mettre à la disposition des systèmes de supervision [Haffar et al, 2007].

Cet équipement sera donc décrit comme un **'PD'** pouvant contenir plusieurs **'LD'**. Dans ce cas, les **'LD'** représentent les équipements physiques connectés au concentrateur. Par conséquent, il y aura autant de **'LD'** dans le concentrateur que d'équipements physiques (i.e. unité de protection) connectés à ce dernier. Dans cette approche, le concentrateur devient finalement virtuel.

Les **'LN'** constituent les fonctionnalités essentielles de l'équipement (ex: protection, contrôle, etc). Ce sont les experts des deux groupes mondiaux de la distribution d'énergie électrique qui ont choisi ces « **LN** ». La partie 7-4 de la norme IEC 61850 [INTERNATIONAL STANDARD IEC 61850-7-4] range les **'LN'** dans des groupes de fonctionnalité. Quelques-uns d'entre eux sont illustrés ci-dessous :

- Groupe de contrôle **'C'**
- Groupe de protection **'P'**
- Groupe des éléments de coupure **'X'**
- Groupe d'acquisition des mesures **'M'**
- Groupe Système **'L'**

Chaque groupe contient plusieurs **'LN'**. Par exemple, 28 fonctionnalités, [Haffar et al, 2007] comme la protection surintensité (**P**ro**T**ection **O**ver **C**urrent **PTOC**), la protection surtension (**P**ro**T**ection **O**ver **V**oltage **PTOV**), et la protection excès de fréquence (**P**ro**T**ection **O**ver **F**requency **PTOF**) [INTERNATIONAL STANDARD IEC 61850-7-4] existent dans le groupe de protection.

Les **'DO'** forment un ensemble de données, regroupées dans une même structure et implémentées en tant que partie des fonctionnalités de l'équipement (i.e. LN). En utilisant l'exemple du groupe de mesure symbolisé par la lettre **M**, une de ces fonctionnalités est le **LN MMXU** qui permet le traitement des valeurs physiques mesurées. Parmi les « **DO** » appartenant à ce « **LN** », il peut y avoir les **'TotW'**, regroupant les données relatives à la mesure de l'énergie, le **'Hz',** regroupant des données relatives à la mesure de la fréquence, le **'A'**, regroupant des données relatives à la mesure de l'intensité du courant électrique, [Haffar et al, 2010b] etc.…

Au final, les **'DA'** constituent les données élémentaires et indivisibles d'une structure d'objets IEC 61850. Le « **DA** » peut avoir un type basique (e.g. int, float, bool) ou bien un type **DAType** défini par la norme IEC61850 [Apostolov et al, 2003].

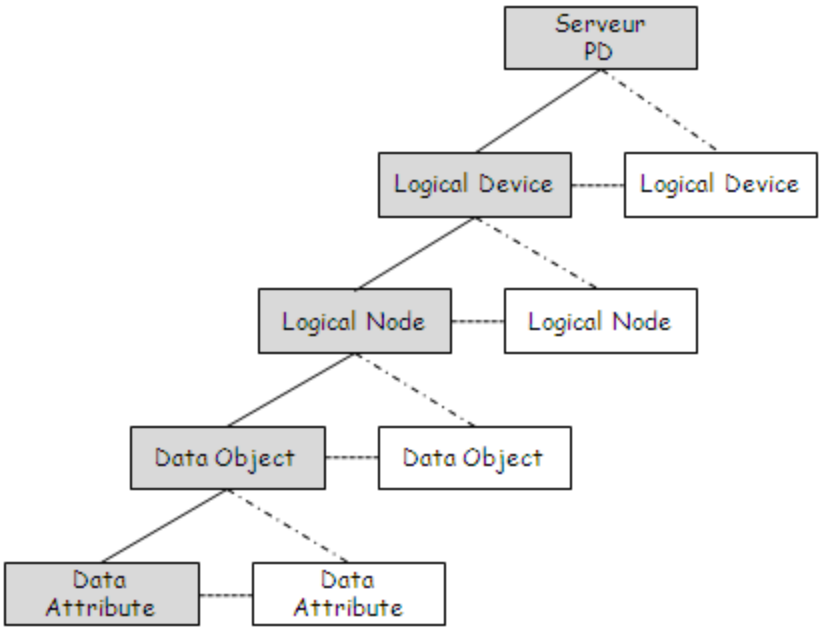


Figure 3.1 Hiérarchie des objets IEC 61850

Excepté les notions d'objet et d'interopérabilité, le standard IEC 61850 apporte une nouvelle perspective {chapitre 3, paragraphe 3.3}, inhérente aux architectures de communication d'un SDEE. Cette perspective s'insère à la fois dans les services de communication et dans l'aspect de configuration des unités de protection afin de leur permettre de réaliser des automatismes distribués par l'intermédiaire du réseau de communication. Les paragraphes suivants mettent en avant le principe de ces avantages.

3.4 Les interfaces de services IEC 61850

La norme IEC 61850 divise l'architecture de communication en trois niveaux distincts afin de définir des interfaces de services réparties entre ces différents niveaux [INTERNATIONAL STANDARD IEC 61850-1]. Les services de communication appartenant à ces interfaces introduisent un trafic de messages variés sur le réseau de communication.

Une exigence temporelle est accordée à chacun de ces types de trafics. [Skendzic et al, 2006] explique que les messages de commande des éléments de coupure, qui sont eux-mêmes échangés entre les unités de protection, exigent des performances temporelles plus contraignantes que les messages dédiés à l'enregistrement des oscillogrammes échangés entre les unités de protection et les systèmes de supervision.

La figure 3.2 montre la répartition de ces interfaces entre les différents niveaux de l'architecture de communication.

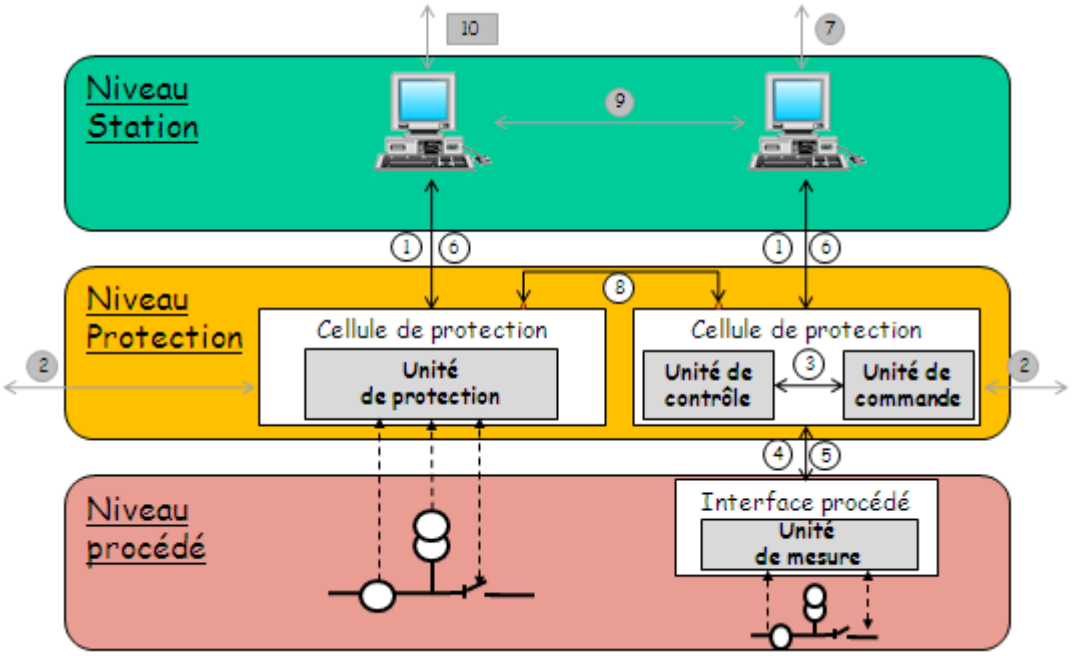


Figure 3.2 Interface de service d'une architecture de communication IEC61850

[Apostolov et al, 2006], [INTERNATIONAL STANDARD IEC 61850-1] illustrent l'explication de chaque interface de services.

Les interfaces grisées 7, 9 et 10 de la figure 3.2 montrent les services échangés au niveau station (i.e. entre les systèmes de supervision). Ces interfaces sont implémentées dans la partie cliente de la norme IEC 61850. D'autre part, l'interface 2 correspond aux services échangés avec des protections distantes ; cette interface ne fait pas partie de la norme IEC 61850.

Etant donné que notre étude se concentre sur les échanges de communication depuis/vers les unités de protection jouant le rôle des serveurs au sens de la communication, les interfaces (1,3,4,5,6 et 8) correspondantes à ce type d'échange sont celles qui seront prises en charge dans notre étude.
L'interface 1 correspond aux services permettant l'échange des données entre les unités de protection et les systèmes de supervision. L'interface 3 contient les services permettant l'échange des données entre les unités de protection et les unités de contrôle appartenant à une même cellule de protection. L'interface 4 illustre les services permettant l'échange des valeurs analogiques entre les unités de mesure et les cellules de protection. L'interface 5 correspond aux services permettant l'échange des signaux de commande entre les cellules de protection et l'interface procédé, composée généralement d'une unité de mesure. L'interface 6 contient les services permettant l'échange des signaux de commande du système de supervision vers les cellules de protection. Finalement, l'interface 8 contient les services permettant l'échange des signaux entre les différentes cellules de protection.

L'interface 1 inclut des services de communication permettant des échanges entre les systèmes de supervision et les unités de protection. Ce type de service existait dans les architectures de communication conventionnelles. Toutefois, l'implémentation des objets dans les unités de protection permet de donner un nouvel aspect à cette interface 1. Cet aspect est illustré par le principe du 'Plug and Play' [Janssen et al, Kema T&D Power], permettant la

découverte automatique du contenu d'un équipement IEC 61850 par une simple interrogation de type **'Auto Discovery'**, envoyée vers l'équipement [Haffar et al, 2010b]. Cet aspect permet de réduire les efforts d'ingénierie au niveau du développement et de la maintenance des systèmes de supervision d'une architecture IEC 61850 [Bruandet et al, 2010].

Les interfaces 4 et 5 contiennent des services de communication utilisés par les SDP, composés de plusieurs unités. Les services de communication appartenant à ces interfaces permettent d'assurer un échange fiable de communication des mesures, entre l'unité de mesure et l'unité de contrôle, et un échange fiable des commandes entre l'unité de commande et l'unité de mesure.

Finalement, l'interface 8 est celle qui va avoir le plus d'intérêt pour notre étude. En effet, cette interface est celle qui permet l'échange de communication entre les équipements appartenant au niveau protection dans le but de leur permettre de réaliser les protections basées sur des automatismes distribués [Apostolov et al, 2006]. Ce type de services n'existait pas dans les SDC basés sur des protocoles conventionnels.

3.4.1 Les profils de communication IEC 61850

Plusieurs profils de communication sont fournis par le standard afin d'assurer le mapping des services IEC 61850 vers les couches standard du modèle OSI.

Les services de communication sont définis par un type d'abstraction intitulée **ACSI** '**A**bstract **C**ommunication **S**ervice **I**nterface' et ce, dans le but de faciliter leur utilisation au niveau des couches applicatives. En effet, les programmeurs de la couche application IEC 61850 peuvent avoir recours à ces services, sans même avoir la moindre connaissance des profils de communication utilisés par ces services [Lloret et al, 2007].

L'ACSI contient les services essentiels suivants : **MMS** '**M**essage **M**anufacturer **S**pecification', les services **GOOSE** '**G**eneric **O**bject **O**riented **S**ubstation **E**vent', les services **SV** '**S**ampled **V**alue' et les services de synchronisation horaire [Engler et al, 2004].

Une interface appelée **SCSM** '**S**pecific **C**ommunication **S**ervice **M**apping' permet le mapping du service vers les couches basses de communication correspondantes [Lloret et al, 2007].

Les fonctions de communication appartenant au service **GOOSE** sont celles qui permettent la réalisation des fonctionnalités de l'interface 8. Cette communication a un caractère de criticité temporelle, ces fonctions sont mappées directement par la SCSM sur les couches basses ISO/IEC 8802-3, court-circuitant ainsi le stack TCP/IP [INTERNATIONAL STANDARD IEC 61850-7-1].

Les fonctions de communication appartenant au service **SV** permettent d'assurer les fonctionnalités des interfaces 4 et 5. Ce service est de nature critique et ses fonctions de communication sont de même mappées directement sur la couche basse ISO/IEC 8802-3 [INTERNATIONAL STANDARD IEC 61850-7-1].

En contrepartie, les fonctions de communication appartenant au service MMS ne possèdent aucune criticité temporelle. Ces fonctions permettent l'échange des communications

conventionnelles de type lecture/écriture entre les unités de protection et les systèmes de supervision. Le SCSM permet donc le mapping de ces services vers les couches TCP/IP standard [INTERNATIONAL STANDARD IEC 61850-1].

Les fonctions de communication appartenant au service **synchronisation horaire** permettent d'assurer la synchronisation des horloges des équipements. Cette communication est mappée par le SCSM vers les couches basses UDP/IP.

Les services Synchro horaire et MMS permettent de pourvoir aux fonctionnalités des interfaces 1 et 6.

Dans la figure 3.3, les services de communication sont répartis en deux parties permettant de pourvoir un SDC intégrant à la fois des échanges d'informations de sécurités (i.e. part 1) et des échanges des informations de fonctionnement générales (i.e. part 2). Ce système est intitulé SDC sécuritaire.

Les services sont mappés sur les couches de communication standard ISO/IEC 9506-1, ISO/IEC 9506-2 et ISO/IEC 8802-3 comme montrés sur la figure 3.3.

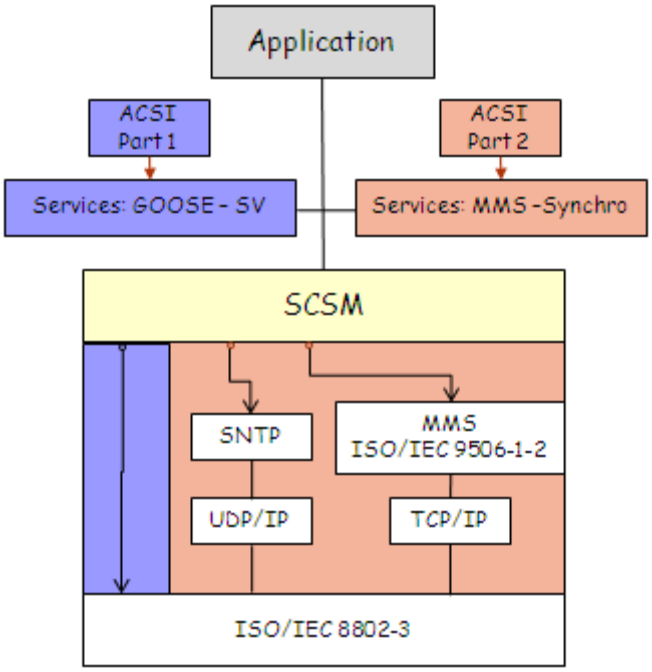


Figure 3.3 Mapping des services IEC 61850

3.5 Configuration d'un SDC IEC 61850

Les nouvelles fonctionnalités de communications existant au niveau des différentes interfaces de services d'un SDC IEC 61850 {chapitre 3, paragraphe 3.2.3}, imposent une configuration de la section communication des unités de protection. Pour cela, à la différence des anciens SDC basés sur des protocoles conventionnels, les unités de protections d'un SDC IEC 61850 sont perçues comme étant divisées en deux sections distinctes configurables par l'utilisateur [Bruandet et al, 2010]:

- La section fonctionnelle composée des paramètres de protection de l'unité en question

- La section de communication permettant de configurer les différents flux de données IEC 61850 échangés depuis/vers l'unité de protection

La norme IEC 61850 impose une configuration standard décrite dans la partie 6 [INTERNATIONAL STANDARD IEC 61850-6]. Cette configuration est établie par l'intermédiaire d'un logiciel standard intitulé '**S**ubstation **C**onfiguration **L**anguage' SCL. Ce logiciel permet de générer les différents fichiers de configuration conformes aux parties 5 et 7 du standard IEC 61850 [INTERNATIONAL STANDARD IEC 61850-5] [INTERNATIONAL STANDARD IEC 61850-7-x].

Cet éditeur n'est pas spécifique à un constructeur particulier du marché, par conséquent, il est capable de manipuler n'importe quel fichier de configuration IEC 61850, indépendamment de sa provenance. Les fichiers de configuration imposés par la norme IEC 61850 sont :

- **ICD** '**I**ED **C**apability **D**escription': fichier de description du contenu d'une unité de protection en terme de fonctionnalités et d'objets conformes au standard IEC 61850. Ce fichier est décrit en format XML conforme à la norme IEC 61850. Il est généralement téléchargeable en ligne sur le site des constructeurs des unités de protection.

- **SSD** '**S**ystem **S**pecification **D**escription': fichier de description de la spécification du SDEE. Ce fichier contient le diagramme unifilaire représentant les différentes structures électriques de l'installation ainsi que les équipements primaires présents dans ces structures (i.e. sources de production, transformateurs, éléments de coupure et jeux de barre).

- **SCD** '**S**ubstation **C**onfiguration **D**escription' : fichier global de configuration du SDC IEC 61850. Ce fichier contient les différentes fonctionnalités de protection (i.e. unités de protection) associées à chaque équipement primaire du fichier SSD ainsi que les interconnexions logiques pouvant exister entre ces fonctionnalités.

- **CID** '**C**onfigured **I**ED **D**escription': fichier de configuration d'une unité de protection IEC 61850. Ce fichier est le résultat d'une configuration complète d'un SDC IEC 61850, il doit être téléchargé dans l'unité de protection.

Afin de réussir la génération des fichiers CID, un échange des fichiers IEC 61850 doit se réaliser entre l'éditeur SCL et les outils de configuration des unités de protection (figure 3.4) [Apostolov et al, 2010a]. L'une des premières phases assurée par l'outil SCL consiste à

importer tous les fichiers ICD ainsi que le fichier SSD dans le but de les fusionner pour créer le fichier de configuration global du SCD. Les outils de configuration des unités de protection interviennent dans la deuxième phase de cet échange (figure 3.4). Leur but essentiel consiste à importer le fichier SCD pour le décomposer par la suite en plusieurs fichiers de configuration unitaires CID qui peuvent avoir un format propriétaire ou standard 61850 [Tan et al, 2009]. Les CID sont les fichiers à télécharger dans les unités de protection afin de créer les deux sections essentielles de ces unités (i.e. la section fonctionnelle et la section de communication) [Bruandet et al, 2010]. La configuration élémentaire de chaque unité de protection aboutit à la configuration globale du SDC IEC 61850 [Goraj et al, 2006].

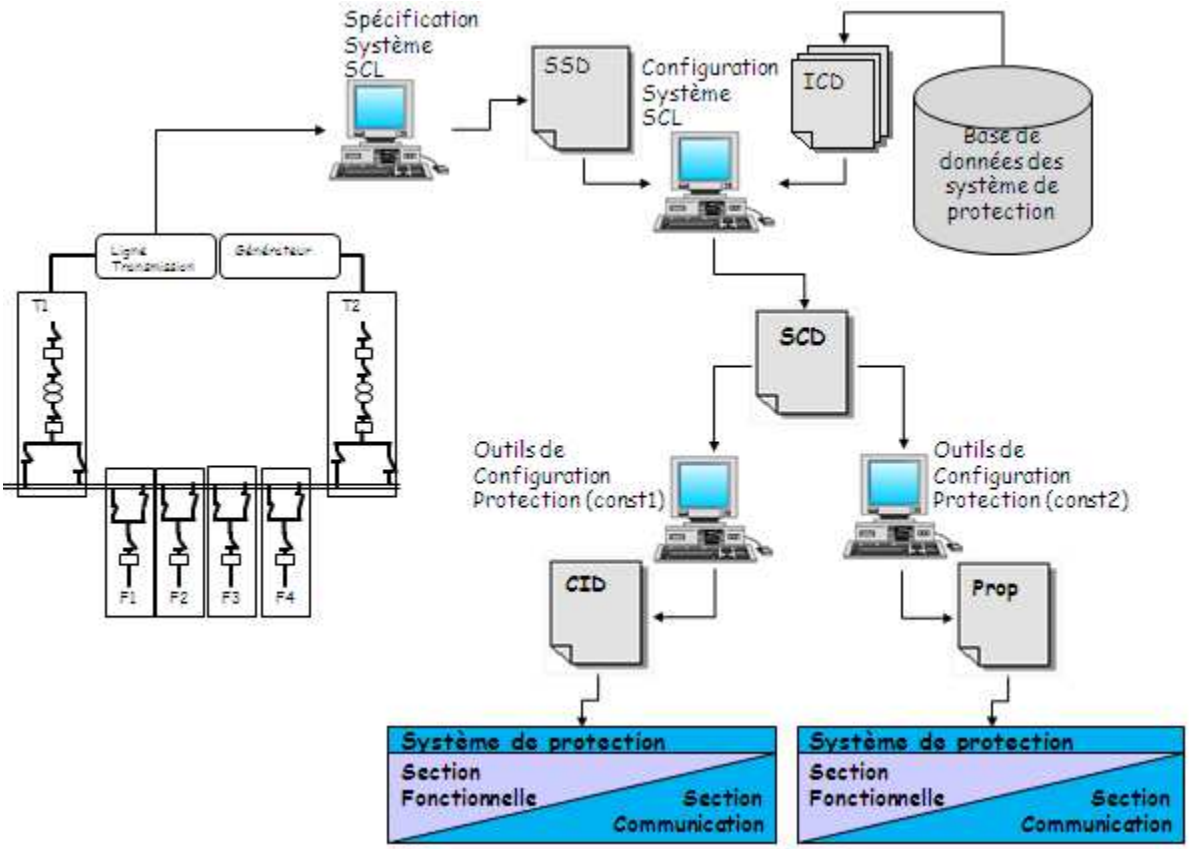


Figure 3.4 Principe de configuration d'une architecture IEC 61850

3.5.1 Flux de données dans une architecture de communication IEC 61850

A la différence des architectures conventionnelles qui disposent d'un seul flux pour l'échange de données, une architecture de communication IEC 61850 en a trois (figure 3.5). Ces flux apparaissent à l'issue d'une configuration totale d'un SDC IEC 61850 (i.e. de toutes les unités de protection de cette architecture et les systèmes de supervision).

Les traits plein de la figure 3.5 symbolisent les flux de données conventionnels échangés entre les unités de protection et les systèmes de supervision. Ces flux existent dans un SDC conventionnel et dans un SDC IEC 61850.

Deux autres flux, en pointillé et en tiret, se rajoutent pour un SDC IEC 61850. Le premier consiste en un échange de communication entre un client et un serveur mais à l'initiative du

serveur. Le deuxième flux intitulé « flux de données inter-équipements » consiste en un échange de données entre les serveurs (i.e. unités de protection).

.

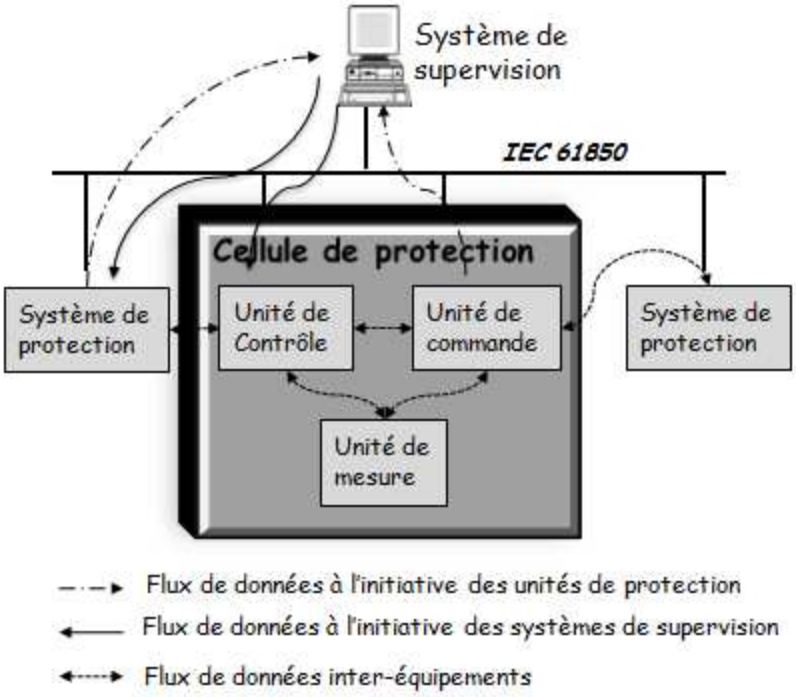


Figure 3.5 Flux de données IEC 61850

3.6 Automatismes distribués à base de messages Goose

Les avantages des protections distribuées basés sur le réseau de communication par rapport aux systèmes traditionnels ayant une connexion câblée point à point entre les unités de protection sont indiscutables.

Ne citons donc que quelques-uns d'entre eux [Barger Pavol 2003]:

- réduction du câblage,

- augmentation de la modularité, flexibilité, re-configurabilité,

- potentiel de communication des informations sur la supervision ou le diagnostic.

Néanmoins, la réalisation des protections distribuées par l'intermédiaire du réseau de communication complexifie la conception et les tests de ces protections.

Ce paragraphe présente le rôle des Goose IEC 61850 et les exigences appliquées à ces signaux, lorsqu'ils sont utilisés comme moyens d'échange de l'information de protections distribuées.

3.6.1 Principe de fonctionnement des messages Goose

Les Goose sont des services fournis par le standard IEC 61850 afin d'assurer des échanges de communication inter-équipement rapides 'Temps Réel' et fiables [Zhang et al, 2008].

Les émetteurs des messages Goose sont appelés publieurs tandis que les récepteurs de ces signaux sont nommés abonnés. Deux caractéristiques principales font la rapidité de ce type d'échange de communication :

- Simplification des protocoles de communication contenus dans les messages Goose
- Attribution d'une priorité élevée pour la transmission de ces messages Goose sur le réseau de communication Ethernet

La première caractéristique provient du fait que les services Goose utilisent un profil de communication court-circuitant la couche transport et la couche réseau. Ainsi, les délais associés aux traitements des acquittements, de segmentation et de réassemblage sont éliminés au niveau publieur et abonné des messages Goose. De plus, cette simplification permet de réduire les délais dus au routage de ces messages au niveau des commutateurs de l'architecture de communication IEC 61850.

La deuxième caractéristique permet de donner la priorité à la transmission de ces messages utilisée au niveau des commutateurs réseaux. Ainsi, ces messages ne feront plus la queue dans les commutateurs. La notion de priorité est effectuée par le remplissage de 3 bits appartenant au champ 'Tag' de l'entête du standard Ethernet. Ces bits permettent de définir 7 niveaux de priorité pour le message Ethernet. Une architecture de communication de type Ethernet commuté est une condition nécessaire pour bénéficier de cette notion. Les commutateurs utilisés dans cette architecture permettent de gérer la notion de priorité.

Le protocole permettant la prise en charge des aspects de priorité est appelé l'IEEE 802.1 p/q [Skendzic et al, 2006]. Ce protocole permet de définir plusieurs files pour l'empilement des messages entrants et chacune de celles-ci possède un niveau de priorité [Zhang et al, 2005]. Les messages occupant la file haute priorité sont transmis avant ceux appartenant aux files de priorités plus basses. Cette configuration est appelée 'Strict Priority' [Skendzic et al, 2006].

Plusieurs études ont été menées de manière à évaluer la performance des messages Ethernet multi-priorités, elles montrent que les délais de transfert des messages à hautes priorités sont plus courts que ceux des messages ayant une priorité moins élevée. Toutefois, la valeur de ces délais dépendra de plusieurs paramètres réseaux dont la capacité des commutateurs [Kakanakov et al, 2007], le nombre d'éléments de l'architecture de communication et les flux de données partageant cette architecture [Zhang et al, 2005].

Les messages Goose sont envoyés en mode multicast par un publieur et sont reçus par tous les équipements appartenant au réseau de communication de ce dernier. Ces messages contiennent des informations indiquant le changement d'état produit au niveau du publieur. En fonction de l'information reçue, les abonnés lancent la procédure de contrôle appropriée et configurée au préalable [Nguyen-Dinh et al, 2007].

Les applications de contrôle associées à ce type d'échange sont, par nature, généralement critiques [Nguyen-Dinh et al, 2007] permettant ainsi la commande des équipements primaires du SDDE. Ceci implique donc que ces signaux doivent assurer, mis à part l'aspect de transfert temps réel entre les équipements, une fiabilité de transmission importante [Hakala-Ranta et al, 2009].

La partie 8.1 de la norme IEC 61850 définit un mécanisme de retransmission des messages Goose permettant d'atteindre un niveau de fiabilité important pour ce type d'échange d'informations [INTERNATIONAL STANDARD IEC 61850-8-1].

Ce mécanisme consiste à établir des retransmissions de ces messages en présence ou en l'absence des évènements produits au niveau du publieur. La retransmission en l'absence des évènements permet aux abonnés de vérifier périodiquement la présence des messages Goose et de signaler une erreur après une perte de communication avec le publieur [Apostolov et al, 2010].

Un champ intitulé **'TimeAllowedToLive'** existe dans chaque message Goose envoyé par les publieurs sur le réseau de communication. Ce champ indique aux abonnés le temps d'attente avant la réception d'un nouveau message Goose provenant du même publieur. La valeur de ce champ est égale au double de la valeur de retransmission du signal par le publieur (i.e. TimeAllowedToLive = 2T0 pour une retransmission stable). Passé ce délai, les abonnés doivent indiquer aux utilisateurs (i.e. système de supervision) une erreur de perte de communication Goose.

La figure 3.6 illustre le principe du mécanisme de retransmission de ces messages [Apostolov et al, 2010].

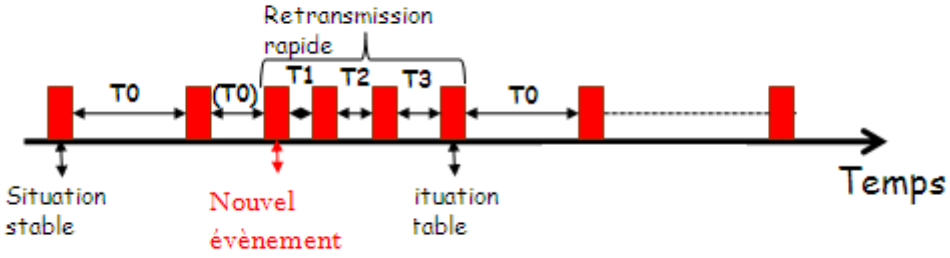


Figure 3.6 Diagramme de retransmission des messages Goose

Quatre paramètres temporels, indiqués sur la figure 3.6, permettent de gérer le mécanisme de retransmission.

Le paramètre T0 est une valeur configurable, indiquant aux publieurs le temps de retransmission du message Goose dans la situation stable de l'équipement (i.e. sans l'apparition d'un changement de statut contribuant à un envoi Goose évènementiel). La valeur de ce paramètre est généralement fixée dans un laps de temps de l'ordre d'une dizaine de secondes [Hakala-Ranta et al, 2009] afin de ne pas charger le réseau de communication.

Suite à l'apparition d'un évènement, le paramètre de retransmission T0 se raccourcit pour l'envoi direct du message évènementiel. Ceci est illustré par le paramètre (T0) sur la figure 3.6.

Une retransmission rapide aura lieu par la suite de cette apparition. La première retransmission s'effectuera après un temps T1 de l'ordre de 3 millisecondes spécifié dans le

paramètre T1. Deux autres retransmissions ralenties auront lieu après des temps T2 et T3 qui sont de même de l'ordre de quelques millisecondes.

Cette retransmission a pour but d'assurer la réception du signal Goose par les abonnés.

Après un phénomène de retransmission évènementiel, la période de retransmission se resynchronise sur l'horloge de T0.

3.6.2 Applications d'automatismes à base de signaux Goose

Plusieurs applications peuvent être réalisées par l'intermédiaire des signaux Goose, elles s'imbriquent d'ailleurs dans le monde des protections basées sur des automatismes distribués.

[WSCC Telecommunications et al, 2001] divisent ces applications en deux catégories, en fonction du type d'information contenue dans le message Goose. En effet, un message Goose contient 64 bits de données configurables par l'utilisateur. Ces données sont de type binaire ou analogique [IEEE PSRC H6: SPECIAL REPORT, 2005].

[WSCC Telecommunications et al, 2001] définissent trois types d'applications d'automatismes distribués pour les messages Goose implémentant les données binaires (i.e. tout ou rien) :

- La première s'appelle Transfert Direct de Commande ou **'Direct Transfer Trip'** ; elle consiste à commander l'ouverture d'un élément de coupure suite à la réception d'un signal Goose. Cette application est généralement utilisée par les systèmes de supervision pour un envoi rapide des commandes urgentes.

- La deuxième application consiste en la sélectivité expliquée dans {chapitre 2, paragraphe 2.4.1}, l'une des applications principales des protections distribuées.

- La troisième est intitulée **'Permissive'**, elle consiste à commander l'ouverture d'un élément de coupure après détection d'une faute locale et réception d'un signal de confirmation envoyé par un autre élément de surveillance des défauts ; [WSCC Telecommunications et al, 2001] ce signal doit être envoyé via les communications Goose et ce, pour ne pas ralentir la prise en compte de la décision de contrôle en cas de défaut.

[IEEE PSRC H6: SPECIAL REPORT, 2005] ajoute que les protections de délestage/relestage et le principe de vote font aussi partie des applications typiquement dédiées à l'utilisation des messages Goose IEC 61850.

[Skendzic et al, 2006] précise que la protection distribuée lors de la défaillance des éléments de coupure peut aussi être établie par l'intermédiaire des messages Goose. Les auteurs introduisent une nouvelle application critique qui peut être accomplie par l'intermédiaire des signaux Goose. Cette application consiste en la protection des bus de distribution électrique. En fait, le principe est le suivant : lors de la détection d'un défaut par l'unité de protection du bus de distribution, un signal Goose sera envoyé par cette même unité à toutes les unités de protection de tous les départs situés en aval. Les unités de protection en aval répondent par des messages Goose indiquant s'ils ont détecté le défaut ou non. Si ces éléments ne détectent

aucun défaut, alors l'unité de protection du bus de distribution prendra la décision de déclencher l'élément de coupure en amont de façon à isoler l'alimentation électrique sur le bus.

Toutefois, l'implémentation des applications citées ci-dessous demeure, encore aujourd'hui, une supposition théorique. Les architectures de communication basées sur le standard IEC 61850 n'ont pas encore pris l'initiative de lancer cette implémentation. Cette méfiance demeure compréhensible au vu de l'aspect non déterministe de l'approche des messages Goose, empêchant, de fait, la comparaison de ses performances avec les approches conventionnelles filaires.

L'étude faite par [Hakala-Ranta et al, 2009] permet de comparer les performances des signaux Goose dans une architecture simple basée sur deux unités de protection, toutefois, dans une architecture réelle, le nombre des unités de protection peut atteindre la centaine. De plus, un phénomène d'avalanche des signaux haute priorité Goose pourra se produire dans une architecture réelle. Ce phénomène contribue à la perte de l'aspect déterministe de ces signaux. Dans une telle situation, l'évaluation de performances des signaux Goose sera une tâche difficile et nécessitera des outils avancés [Bruandet et al, 2010].

Pour illustrer ce phénomène, prenons l'exemple de l'architecture de la figure 3.7. Dans cette architecture, deux défauts électriques se produisent au même instant, au niveau du départ D1 et du jeu de barre J1.

En partant de l'hypothèse que les automatismes distribués sont réalisés par l'intermédiaire des signaux Goose dans cette architecture, des échanges de communication temps réel seront lancés avec un rythme d'échange évènementiel entre l'unité de protection de jeu de barre J1 et les unités de protection des départs liés à ce jeu de barre et ce, dans le but de confirmer l'isolation électrique du jeu de barre.

Au même instant, des communications temps réel seront échangées entre l'unité de protection du départ D1 et l'unité de protection en amont de celui-ci pour assurer la sélectivité de cette partie de l'installation.

A l'issue de l'isolement de l'alimentation de J1 et de la sélectivité de la partie de l'installation en D1, les messages Goose gardent le rythme de retransmission évènementielle exigée par la norme de façon à assurer la fiabilité de cette transmission.

A ce moment-là, un basculement vers une production électrique interne sera produit au niveau du tableau 2 de la figure 3.7. Ce basculement entraîne la mise en route de la procédure de délestage/relestage des charges pour assurer une consommation équivalente à la puissance délivrée par les groupes. Une unité de protection conventionnelle, numérotée 81 par (ANSI/IEEE C37.2-1979), est celle qui prend en charge le traitement de ce genre d'automatisme en envoyant des signaux Goose vers les unités de protection des départs pour délester les consommateurs non prioritaires de ce tableau.

En conséquence, une avalanche de Goose sera produite au niveau du réseau de communication IEC 61850. Cette avalanche introduira un doute au niveau des performances temporelles des signaux Goose envoyés.

De fait, l'implémentation pratique des applications d'automatismes, distribuées à partir des signaux Goose, nécessite une évaluation pointue des performances de ces signaux. Une étude

comparative entre les performances de ces signaux de communication et les performances des signaux filaires doit être menée avant la mise en place d'une telle solution [Hakala-Ranta et al, 2009], [Zhao et al, 2009].

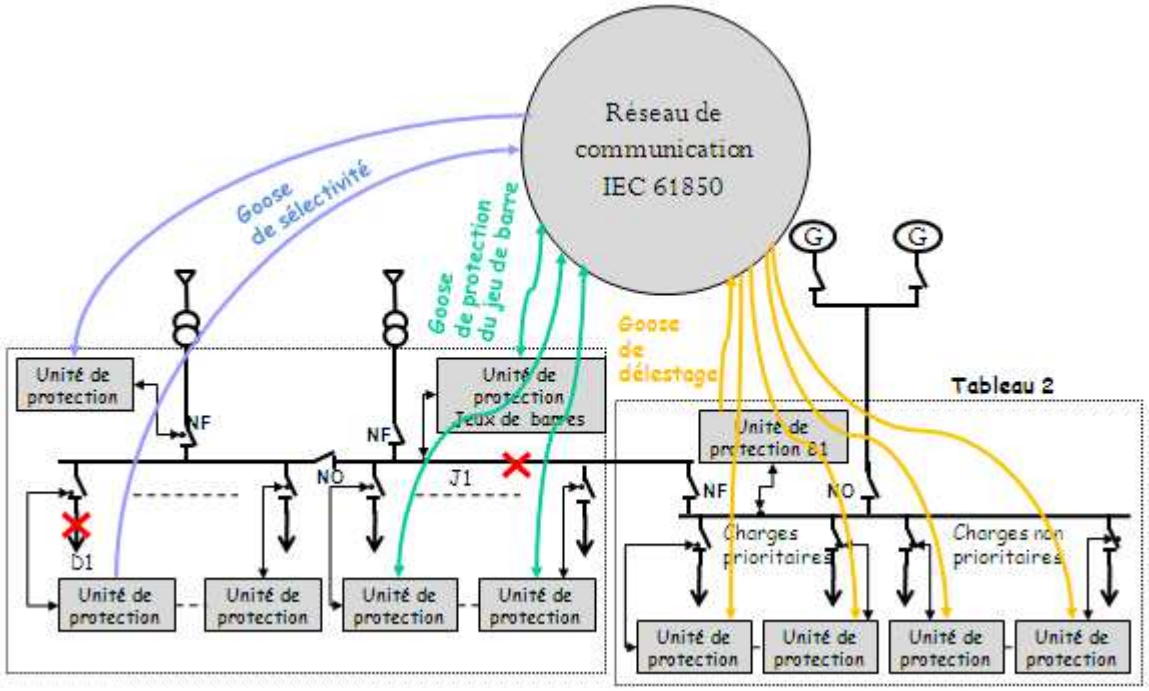


Figure 3.7 Exemple d'avalanche des messages Goose

3.7 Conclusion

Le nouveau standard IEC61850 a ouvert la possibilité d'échanger plusieurs types de flux de données dans un même SDC (système de communication) d'un SDEE (système de distribution d'énergie électrique).

Cette notion autorise la construction d'une architecture interopérable dans laquelle la réalisation de protections distribuées sera effectuée par l'intermédiaire du réseau de communication.

Parmi ces flux il existe ceux qui possèdent un caractère sécuritaires ; ils sont utilisés pour les applications distribués entre les systèmes de protection. Des contraintes temporelles sont par conséquent appliquées à ce genre de flux de communication.

L'évaluation et la vérification de performance d'un système de communication IEC 61850 demeure ainsi une tache essentielle à réaliser lors de la mise en place de n'importe quelle architecture de communication à base de ce standard.

Cette évaluation doit prendre en compte les pires situations du réseau de communication dans lesquelles une avalanche de ces flux pourra se produire.

Chapitre 4

Approche pour la vérification d'un système de communication IEC 61850

4.1 Introduction

L'autorisation des échanges critiques des messages a rendu du SDC IEC 61850 un élément essentiel de la sûreté de fonctionnement d'un SDEE. Les différents moyens de la SDF, tels que la vérification et l'évaluation de performance, doivent par conséquent être appliqués à ce système.

Le paragraphe 2 décrit les paramètres de performances et les exigences normatives à respecter pour les différents flux de communication IEC 61850.

Les approches habituellement utilisées pour établir des études de performances des systèmes de communication sont répertoriées dans le paragraphe 3. Ce paragraphe focalise sur les évaluations de performance des architectures à base du standard IEC 61850.

Le paragraphe 4 rend compte des vérifications à réaliser dans le cadre du moyen d'évitement des fautes. Nous démontrons à la fin de ce paragraphe que l'approche par Co-Simulation est celle la plus appropriée pour la réalisation des vérifications d'une architecture IEC 61850.

Finalement, le dernier paragraphe explique l'effet de la Co-Simulation sur les moyens de la sûreté de fonctionnement d'un SDC IEC 61850.

4.2 Paramètres de performance des messages Goose

Les caractéristiques de la performance des messages de communication IEC61850 sont décrites dans plusieurs articles [Atienza et al, 2010], [INTERNATIONAL STANDARD IEC 61850-6], [Xyngi et al, 2010]. Ces articles définissent deux paramètres essentiels pour décrire la performance des messages de communication IEC 61850 : le temps de transmission et le temps de transfert (figure 3.8).

Le temps de transmission est composé de trois constituants distincts (figure 3.8):

- Ta : Temps du traitement du message par l'interface de communication du publieur
- Tb : Temps de propagation du message au niveau du réseau de communication
- Tc : Temps de traitement du message par l'interface de communication de l'abonné

Ce temps est reconnu dans d'autres articles [WSCC Telecommunications et al, 2001] sous le nom de délai de bout en bout. Selon les auteurs, ce paramètre peut, à son tour, être divisé en trois catégories pour les applications de protection :

- Le délai de bout en bout maximal. Ce paramètre représente le temps maximal autorisé pour une communication transmise
- La variation du délai de bout en bout. Ce paramètre représente la valeur maximale autorisée pour la variation du paramètre de délai

- L'inégalité du délai de bout en bout. Ce paramètre représente la valeur maximum de l'écart entre le délai bout en bout d'envoi et le délai de bout en bout de réception

Le temps de transfert constitue le délai pour lequel un récepteur réagit suite à la réception d'un message de communication Goose. La réaction du récepteur se traduit par le lancement de l'opération de contrôle correspondant au message Goose reçu. Ce temps est calculé en additionnant le temps de transmission du message Goose et le temps de réaction du récepteur de ce message (figure 3.8).

La norme IEC 61850 ne met pas l'accent sur les catégories de délai de bout en bout [INTERNATIONAL STANDARD IEC 61850-6]. Le paramètre retenu par la norme est la valeur maximale du temps de transmission (i.e. délai de bout en bout).

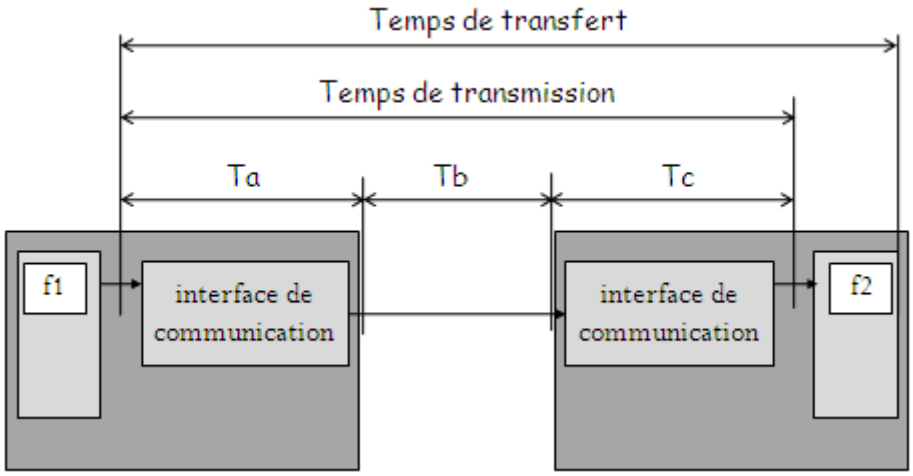


Figure 4.1 Délai de transmission et de transfert des messages de communication

4.2.1 Exigences normatives à respecter

La norme IEC 61850 divise les services de communication en 7 types différents dans le but de les organiser dans des classes de performances spécifiques (tableau 3.1). A leurs tours, les classes de performances sont divisées en deux catégories ; M et P ; et sont définies en fonction des fonctionnalités d'un SDEE et du type de service. Ces classes sont considérées comme indépendantes de la taille et du nombre d'équipements de l'architecture de communication du système. La caractéristique essentielle d'une classe de performance est la valeur maximale du temps de transmission (i.e. délai de bout en bout du transfert de service).

La classe M correspond aux performances des messages de surveillance et de gestion de la qualité d'énergie. La classe P correspond aux performances des messages de contrôle et de protection envoyés par le biais du réseau de communication. Cette dernière est divisée en trois parties, P1, P2 et P3. La partie P1 correspond aux performances des messages de contrôle nécessitant de faibles contraintes temporelles, la partie P2 correspond à ceux qui n'ont pas d'exigences temporelles spécifiées et enfin, la partie P3 correspond aux performances des messages exigeant de fortes contraintes temporelles.

Le type 1 représente les messages contenant des informations binaires qui sont dédiées à l'échange des commandes ou des informations critiques par le biais du réseau de

communication. Le temps de transmission est un paramètre critique pour ce type de service. Le déclenchement, le blocage, le ré-enclenchement font partie des commandes de ce type, tandis que le transfert des changements d'état fait partie des informations critiques. A l'issue de la réception des services du type 1, les unités de protection doivent réagir immédiatement en lançant la fonctionnalité interne adéquate. Ce type de service est typiquement dédié aux interfaces des services 3, 5 et 8, expliquées au début de ce chapitre. La norme divise ce type en deux catégories ; le 1A qui correspond au transfert de messages de contrôle direct au travers du réseau et le type 1B, moins contraignant, correspond à l'utilisation de ce service critique pour d'autres transferts (i.e. pas pour un échange des messages de contrôle). Les types 1A et 1B peuvent être attribués aux classes P1 P2 ou P3 en fonction de l'application d'automatismes distribués qui les utilise.

Le type 2 correspond aux services pour lesquels le temps de génération constitue un paramètre de criticité plus fort que le temps de transmission. Les messages de communication appartenant à ce type sont généralement horodatés à la source (i.e. le temps de génération est implémenté par l'émetteur dans le message de communication). Les récepteurs doivent réagir en fonction de la valeur de la variable temporelle incluse dans le message reçu.

Le type 3 correspond aux messages dédiés à la télécommande, télésignalisation, télémesure, télé contrôle et transmission des alarmes. Ces messages peuvent être de type horodaté comme dans le cas de transmission des alarmes ou non horodatés dans les autres cas. Les échanges des rapports de données constituent un des services du type 3. Cet échange peut être de nature évènementielle (i.e. suite à un changement d'état de l'un des constituants du rapport) ou périodique.

Le type 4 correspond au transfert des valeurs analogiques soit entre les unités de mesures et les unités de contrôle soit entre les unités de contrôle et les unités de protection. Les messages de communication appartenant à ce type sont dédiés aux interfaces des services 4 et 8 de la norme IEC 61850.

Le type 5 correspond au transfert des fichiers du type : les fichiers d'enregistrements, les fichiers de configuration etc. Ce type de transfert est caractérisé par la plus faible contrainte temporelle de tous les types.

Le type 6 représente les messages de synchronisation horaire. Aucune exigence n'existe au niveau de la performance temporelle de ces messages à l'exception de l'exactitude de la synchronisation des équipements.

Tableau 4.1 Types et performances des messages IEC 61850

Type	Applications	Classe performances	
1A	Applications critiques de déclenchements des éléments de coupures	P1 P2/P3	10 ms 3ms
1B	Autres transmissions rapides moins contraignantes	P1 P2/P3	100 ms 20 ms
2	Messages horodatés à vitesse		100 ms

	moyenne		
3	Messages lents de télégestions		500 ms
4	Transfert valeurs échantillonnées	P1 P2/P3	10 ms 3ms
5	Transfert lourd		> 1000 ms
6	Synchronisation horaire		Non défini

Le critère de la performance est exprimé par l'intervalle de la valeur du temps de transmission des différents types de services. Ce critère est illustré au niveau du tableau ci-dessous [INTERNATIONAL STANDARD IEC 61850-5], [Xyngi et al, 2010]. La valeur de ce temps de transmission peut aller de l'ordre de 3 ms (i.e. pour les messages de type 1A) jusqu'à plusieurs secondes pour les messages de transfert des fichiers lourds.

4.3 Evaluation des performances des réseaux de communication IEC 61850

L'implémentation des automatismes de protections distribués en utilisant les fonctionnalités de communication avancées du standard IEC 61850 exige une étude de la fiabilité des échanges de communication. Cette étude vise à vérifier le respect des contraintes temporelles, l'amélioration des performances et la minimisation des coûts [Caufriez Laurent, 2005]. Par conséquent, le terme 'fiabilité de communication' vient dans le sens de la vérification de l'intégrité de l'information en terme d'une étude des défaillances temporelles pouvant impacter cette information. La fiabilité de communication prise de ce point de vue est définie par la viabilité de l'information de communication [Caufriez Laurent, 2005]. Une information est considérée viable lorsqu'elle peut se mettre à disposition de l'exploitant durant une certaine durée de vie. Au-delà de ce laps de temps, l'information sera jugée obsolète et ne sera plus exploitable (i.e non viable).

Par conséquent, durant cette étude, des comparaisons préalables (i.e. phase initiale du développement du SDEE) entre les performances des échanges de communication IEC61850 et les exigences normatives imposées pour chaque échange doivent être établies. Cette comparaison peut de même être conduite entre les performances des échanges de communication IEC 61850 et les exigences temporelles des applications d'automatismes distribués. Pour ces raisons, une évaluation pointue de performance des échanges de communication doit être réalisée, en tenant compte de la globalité des paramètres de l'architecture de communication.

Cette étude est la plus importante à mener tout au long des phases de développement d'un nouveau SDC IEC 61850. La fiabilité des échanges de communication, en particulier, doit être évaluée lors de la phase de conception, et ce, dans le but d'apporter un niveau de confiance élevé dans le choix de l'architecture de communication. Ceci permet de guider les clients finaux pour implémenter les fonctionnalités de communication avancées de cette nouvelle technologie.

L'étude de la fiabilité des échanges de communication doit être réalisée selon plusieurs architectures de communication qui sont proposées. Le passage d'une architecture vers une autre est réalisé grâce à la reconfiguration de ses paramètres ou à un changement de ses composants constitutifs ou de leur organisation topologique.

En plus de la fiabilité des échanges de communication, [Ali et al, 2008] [Ito et al, 2008] [Chenghong et al, 2008] considèrent que la disponibilité de l'architecture de communication constitue un deuxième paramètre important à prendre en compte. Selon les auteurs, la disponibilité est considérée comme une fonction de la topologie du réseau et du niveau de redondance appliqué aux constituants de l'architecture. En effet, le passage d'une architecture étoile vers une architecture en anneau augmente d'une part la disponibilité de l'architecture de communication mais d'autre part le coût de son installation. Outre le choix de topologie, [Ali et al, 2008] [Thomas et al, 2010] proposent une duplication des commutateurs réseaux dans leur architecture de communication anneau. Les auteurs partent de l'hypothèse que chaque unité de protection IEC 61850 peut intégrer deux interfaces de communication pouvant se connecter chacune sur l'un des anneaux de l'architecture. Leur hypothèse se base sur le fait que les constructeurs ont commencé une telle implémentation de communication dans leurs unités de protection.

Selon [Udren et al], le choix de la structure du réseau de communication constitue la cinquième étape de construction de l'architecture de communication IEC 61850.

L'étape 1 est le choix d'une architecture de communication respectant les différentes normes, liées aux contraintes environnementales et électriques.

Les étapes 2, 3 et 4 sont constituées d'un choix adapté des unités de protection de manière à respecter l'interopérabilité, la conformité à la norme IEC 61850 et l'intégration les fonctionnalités de communication demandées.

L'architecture de communication doit être choisie de manière à pouvoir accommoder les performances exigées dans le pire des cas d'échange de flux de données (e.g. exemple d'une avalanche des rapports de données accompagnée d'une avalanche des messages Goose suite à des évènements électriques produits au niveau de l'installation).

De même, l'architecture doit être capable de prendre en compte des extensions futures (i.e. équipement, ajout de configuration sur les équipements) qui ajoutent un trafic de communication supplémentaire au réseau.

4.3.1 Paramètres affectant les performances d'un SDC IEC 61850

[Zhao et al, 2009] ont retenu dans leurs études les deux paramètres suivants: la topologie du réseau de communication et le nombre d'équipements constituant l'architecture de communication.

Ils proposent deux topologies dans leurs études : la topologie bus et la topologie étoile. Une configuration identique des flux de données a été mise en place pour tous les équipements, de manière à envoyer 1920 messages IEC 61850 de type 4 par seconde.

Le nombre d'équipements est la variante utilisée dans chacune des deux topologies, et ce, dans le but de déterminer la quantité maximale d'équipements pouvant être implémentés, tout en respectant l'exigence normative du service de communication de type 4 (i.e. 3 ms).

Les résultats de leurs travaux montrent qu'en utilisant une telle configuration de flux de données, la topologie de bus peut contenir au maximum 13 équipements tandis que la topologie étoile peut implémenter 53 équipements.

[Ali et al, 2008] ont réalisé une étude de performance d'une architecture de communication IEC 61850 en prenant en compte la contrainte de disponibilité.

Les topologies : étoile, étoile redondante, anneau et double anneau, sont mentionnées dans cet article qui précise que la topologie en double anneau est la mieux adaptée à une architecture de communication IEC 61850. Cette topologie est caractérisée par la plus haute disponibilité.

Une étude de performance a été effectuée sur une architecture de communication basée sur une topologie en double anneau et constituée de 8 cellules de protection. Les deux variantes retenues dans cette architecture sont : la rapidité du réseau de communication (i.e. 10Mbits/secondes, 100Mbits/secondes et 1Gbits/secondes) et le taux des flux de données envoyés sur l'architecture.

Les résultats de performance obtenus montrent que les critères de performance ne sont plus respectés seulement si les flux de données dépassent le taux de 4800 échantillons/seconde pour une architecture de 10 Mbits/sec.

Néanmoins, les auteurs précisent qu'une telle architecture engendre un inconvénient au niveau du prix de par les redondances d'équipements réseau et des interfaces de communication pour les différentes unités appartenant aux cellules de protection.

[Chenghong et al, 2008] précise que l'implémentation de protocoles avancés tel que le VLAN, basé sur le protocole IEEE 802.1q, permet d'atteindre une transmission temps réel et améliore les performances des échanges de communication. Ceci est accompli grâce à la division du domaine de diffusion des messages multicast qui permet de réduire les échanges de flux de données sur des parties de l'architecture de communication.

Ainsi, on peut conclure de tous ces travaux que plusieurs variantes doivent être étudiées lors de l'évaluation de performance d'une architecture de communication. Une coordination efficace entre ces variantes doit être mise en place pour atteindre les critères exigés sur les performances des messages de communications de l'architecture en question.

4.3.2 Approches générales d'évaluation de performances d'un SDC industriel

Plusieurs articles publiés dans diverses conférences permettent d'introduire les différentes approches utilisées pour évaluer les performances des réseaux de communication industriels.

Ces approches peuvent se diviser en quatre catégories : l'approche analytique, l'approche expérimentale, l'approche par simulation et l'approche de Co-Simulation.

4.3.2.1 Approche analytique

Cette approche est généralement utilisée pour évaluer les performances d'un réseau basé sur une architecture de communication simple. L'ordonnancement d'envoi des trames ainsi que les différents flux transitant au niveau de l'architecture de communication doivent être connus au préalable, pour utiliser efficacement cette approche. De plus, les phénomènes de collision doivent être négligeables.

[Robert et al, 2010] propose une comparaison entre les performances de plusieurs architectures industrielles. EtherCat, Profinet IRT, ModBus/TCP et Ethernet/IP sont les protocoles de communication choisis. Le mécanisme d'échange de communication entre les différents composants de l'architecture est basé sur le principe du maître/esclave. Ce mécanisme permet d'éliminer les effets de collision étant donné que l'échange de communication ne se fait qu'à l'initiative du maître de la communication. Un seul maître de communication est considéré pour chacune des architectures. La formule analytique du calcul du paramètre de performance (temps de cycle) est divisée en trois parties essentielles : temps de transmission du message, temps de propagation et temps de latence des équipements. Le temps de transmission dépend de la taille du message transmis mais aussi du débit de l'interface de communication de l'équipement. La latence est définie comme étant le temps d'attente du message dans l'équipement réseau. Enfin, le temps de propagation est considéré comme dépendant de la distance séparant les différents équipements ainsi que de la rapidité du réseau (type des liens de communication).

Une étude similaire a été proposée par [Alessandria et al, 2007] concernant une architecture formée d'un automate programmable avec trois modules d'entrée/sortie déportés. Le paramètre de performance choisi est le temps de transmission. La formule analytique permettant la détermination de ce paramètre ressemble à celle proposée par [Robert et al, 2010]. Leur différence, cependant, consiste en l'ajout d'un paramètre indiquant le temps d'encapsulation et de décapsulation des trames dans la formule proposée par [Alessandria et al, 2007]. Dans leurs études, les auteurs mentionnent que le réseau de communication n'est chargé que par trois échanges de communication qui permettent la commande des sorties de chaque module déporté sur le réseau de communication. Ainsi, la nature des flux de données est bien connue. D'autre part, étant donné que le réseau n'est pas chargé par d'autres échanges de communication, il n'y a aucun risque de collision. Ces deux aspects ont permis l'utilisation de l'approche analytique dans leurs études.

Néanmoins, l'aspect aléatoire des échanges de communication, dû à la notion évènementielle de certains services de communication IEC 61850, empêche la reconnaissance préalable des flux de communication. De plus, les collisions, à l'intérieur d'un tel réseau, peuvent apparaître fréquemment du fait de la possibilité de phénomènes d'avalanche des signaux haute priorité. Toutes ces raisons expliquent que, dans des architectures complexes basées sur des protocoles de communication avancés, l'utilisation de l'approche analytique ne peut être utilisée sans avoir recours à des simplifications, celles-ci pouvant aboutir à des mesures de performance erronées ou imprécises [Pereira et al, 2004]. Pour cela, cette approche demeure inappropriée pour une architecture de communication IEC 61850.

4.3.2.2 Approche expérimentale

Cette approche consiste à réaliser des essais sur la plateforme réelle, constituée du vrai réseau de communication implémentant les équipements de communication et les composants essentiels du réseau (i.e. routeurs, commutateurs, liens etc.).

[Alessandria et al, 2007] ont utilisé cette approche pour évaluer la performance d'un réseau de communication industriel basé sur le protocole Ethernet/IP. L'architecture proposée par les auteurs consiste à mettre en place un automate programmable, équipé des modules d'entrée/sortie locaux eux-mêmes connectés à d'autres modules d'entrée/sortie déportés sur le réseau Ethernet/IP. La méthodologie utilisée pour le calcul du délai de transmission consiste en fait à envoyer au même moment des commandes d'activation des sorties appartenant aux modules locaux et aux modules déportés. Une horloge connectée à ces modules permet de déterminer le temps d'activation des sorties. La différence entre le temps d'activation de ces sorties détermine la performance du réseau de communication qui a été utilisé.

De même, [Kalappa et al, 2006] ont utilisé cette approche pour évaluer les performances d'un réseau de communication Ethernet/IP. L'analyseur réseau est l'élément de base proposé par les auteurs pour cette évaluation de performance.

L'analyseur réseau est un logiciel de capture de trames permettant de donner les détails de tous les messages de communication échangés, par l'intermédiaire de l'interface de communication de l'ordinateur sur lequel il est installé. Parmi les détails fournis par ce logiciel, on cite ceux qui sont les plus importants :

- Détection des protocoles de communication implémentés dans les messages émis ou reçus
- Détermination des temps de transmission et de réception des messages de communication. Ces temps indiquent l'instant de l'envoi ou la réception de la trame de communication complète

Leur étude propose une architecture basée sur des automates programmables qui sont reliés par l'intermédiaire d'un commutateur. L'ordinateur qui implémente l'analyseur réseau est connecté au commutateur reliant les automates programmables, et ce, pour prendre une image des messages échangés entre les automates. Le calcul des paramètres de performance de leur architecture est basé sur les mesures collectées par l'analyseur réseau (Wireshark).

Cette approche est généralement utilisée pendant la phase de test de l'architecture de communication d'un SDEE.

[Hakala-Ranta et al, 2009] présente l'étude de performance d'une simple architecture de communication IEC 61850 formée de deux unités de protection. Le 'Temps de transmission' est le paramètre de performance choisi dans leur étude. La configuration des flux de données a consisté en un échange de 2000 messages Goose/seconde entre les deux unités de protection. Les résultats de performance de 225 tests ont montré que le temps de transmission varie entre 1ms et 4ms.

L'analyse de ces résultats de performance montre que les messages Goose ont respecté le critère de type 1A imposé par la norme (i.e. 3ms). Le dépassement de 3 ms pour certains tests a été expliqué par le fait que cette mesure introduit le temps de traitement au niveau du récepteur (i.e. temps de cycle de l'équipement). Les auteurs quantifient ce temps de traitement entre 0 ms et 2.5ms.

Cet article ne montre pas la méthode utilisée pour le calcul de ce paramètre de performance. Toutefois, le fait que le temps de traitement du récepteur est inclus dans le paramètre de performance nous permet de deviner que le calcul a été fait au niveau du récepteur en se basant sur le principe de l'horodatage. En effet, un message Goose contient un champ temporel indiquant le moment de sa génération. D'autre part, les unités de protection intègrent généralement des procédures d'horodatage des évènements (i.e. que ce soit un évènement comme un changement de statut ou un évènement de réception de communication). À partir du moment où la réception des signaux est horodatée à la destination (i.e. par le récepteur), la performance sera calculée par une simple différence entre le temps de détection du signal Goose et la valeur temporelle contenue dans ce signal, indiquant le temps de sa génération.

[Tan et al, 2009] ont pareillement utilisé cette approche pour l'évaluation de performance d'une architecture de communication IEC 61850 interopérable. Les auteurs se sont basés sur la même méthodologie pour calculer les performances.

Ainsi, dans cette partie ont été mentionnées les différentes méthodologies utilisées pour évaluer les performances d'une architecture de communication réelle utilisant l'approche expérimentale. Pour un calcul précis de l'ordre de la milliseconde, la technique la plus utilisée est celle qui est basée sur le principe de l'horodatage. Cette technique ne nécessite pas la présence d'équipements tiers tels que les analyseurs réseau, mais exige des équipements assez intelligents permettant de gérer cette notion. Le but essentiel de l'horodatage, pour un tel type d'utilisation, revient à réduire les effets temporels dus au traitement de ces équipements tiers qui entraînent généralement des imprécisions au niveau des calculs [Kalappa et al, 2006].

4.3.2.3 Approche de simulation

Cette approche est essentiellement utilisée lors de la phase de conception d'un système de communication dans laquelle les composants constitutifs de ce SDC sont absents. Par conséquent, cette approche autorise la substitution de ces composants par des modèles construits dans l'environnement de la simulation.

Ces modèles doivent inclure une image conforme de la section de communication des composants réels. Pour atteindre cet objectif, vérifier que les protocoles de communication implémentés dans les modèles sont conformes à ceux existant dans les composants réels est une condition nécessaire à respecter.

L'avantage de cette approche est, en réalité, que les modèles puissent être instanciés dans l'environnement de simulation pour atteindre un nombre équivalent à celui des composants de communication de l'architecture réelle.

L'architecture simulée sera vue comme étant une image virtuelle de l'architecture réelle pouvant refléter le même comportement que cette dernière [Pereira et al, 2004]. La génération des flux de données dans l'architecture simulée est faite par changement des attributs des modèles de communication [Pereira et al, 2004].

Le principal avantage de cette approche réside dans le fait de pouvoir générer plusieurs études de performance en effectuant des changements au niveau de l'architecture simulée. Ces changements peuvent être appliqués soit au niveau de la conception de l'architecture simulée (i.e. topologie, liens, types des commutateur/routeurs, utilisation des protocoles avancés type

VLAN etc.) ou soit au niveau de la configuration des flux de communication générés par les modèles de simulation (i.e. modification du nombre de message Goose, reporting etc.).

La réalisation de ces changements nécessite des efforts minimes en comparant cela à une architecture de communication réelle. Ceci constitue l'une des raisons pour lesquelles l'utilisation de cette approche est répandue dans le monde de l'évaluation des performances d'un réseau de communication IEC 61850.

L'étude de performance de l'architecture IEC 61850 réalisée par [Zhao et al, 2009], expliquée dans {chapitre 3, paragraphe 3.4.1}, repose sur une approche de simulation.

[Thomas et al, 2010], [Ali et al, 2008] et [Sidhu et al, 2007] ont proposé la simulation d'une architecture IEC 61850 en utilisant le logiciel de simulation OpNet Modeler. Trois changements essentiels ont été proposés pour la construction des différentes architectures simulées, le temps de transmission était le paramètre de performance évalué pour chacune des architectures simulées. La différence entre les architectures simulées est due aux changements suivants :

- Rapidité du réseau de communication (i.e. trois types de lien réseaux ont été proposés 10Mb/s, 100Mb/s, 1Gb/s)

- Flux de données échangés au niveau de l'architecture simulée (i.e. trois types de flux ont été proposés : 960 valeurs/secondes, 1920 valeurs/secondes et 4800 valeurs/secondes)

- Types de commutateurs réseau (i.e. deux types de commutateurs ont été proposés par les auteurs : un commutateur normal et un commutateur avancé qui gèrent les mécanismes de priorité et de VLAN).

L'étude de performance de l'architecture de communication simulée IEC 61850, et proposée par [Sidhu et al, 2010], est basée sur le changement du paramètre de la gestion de priorité implémenté dans les couches basses (i.e. plus spécifiquement la couche Ethernet) des unités de protection IEC 61850.

Plusieurs mécanismes tels que FIFO, Strict Priority, Round Robin et Weighted Round Robin peuvent être attribués à ce paramètre afin de permettre une gestion efficace de la priorité des messages de communication [Sidhu et al, 2010]. Ces mécanismes sont habituellement implémentés dans les commutateurs réseaux pour leur permettre de définir plusieurs queues d'attente des messages de communication. Le seul mécanisme implémenté dans la couche MAC des équipements de communication IEC 61850 est le FIFO (i.e. First In First Out). Les auteurs proposent une évaluation du temps de transmission des messages sécuritaires Goose en faisant varier le mécanisme de gestion de priorité au niveau de la couche MAC des unités de protection. Le résultat de leur étude montre que la variation du mécanisme de gestion de priorité au niveau des unités de protection a un impact direct sur la performance des signaux Goose. Cette dernière passera de 4,14 ms à 0,5 ms selon les mécanismes choisis.

Les approches expérimentale et analytique ne peuvent pas être utilisées à ce stade étant donné l'absence totale d'équipements et la complexité du réseau de communication [Bruandet et al, 2010]. Toutefois, l'approche de simulation présente des inconvénients majeurs, notamment au niveau du temps requis pour exécuter des simulations et des efforts de programmation demandés pour la construction des modèles d'équipement [Pereira et al, 2004]. Par ailleurs, le

paragraphe 2.5 va montrer que cette approche ne pourra pas être utilisée lors des phases avancées du développement d'un SDEE, en particulier lors de la phase de réalisation des tests associés à l'architecture de communication IEC 61850.

4.4 Approche de Co-Simulation pour les réseaux IEC 61850

L'approche de Co-Simulation peut être considérée comme un mélange entre l'approche de simulation et l'approche expérimentale. Son but essentiel est de pouvoir connecter les modèles de communication conçus au niveau du logiciel de simulation à des équipements appartenant à une plateforme de communication réelle. L'utilisation de cette approche est pertinente dans le cas où les expérimentations ne peuvent pas être réalisées à cause de l'absence d'équipements réels. Par conséquent, ces équipements seront remplacés par des modèles développés dans l'environnement de Co-Simulation. Les modèles de Co-Simulation représentent une image virtuelle du comportement de la section de communication de l'équipement réel. [Hasan et al, 2008] ont utilisé cette approche pour connecter un contrôleur d'un pendule inversé avec des capteurs et des actionneurs via un réseau de communication simulé dans le logiciel OpNet Modeler.

Comme il a été expliqué précédemment dans {chapitre 2, paragraphe 2.3.7.2}, l'aspect de vérification constitue l'un des éléments essentiels du moyen d'évitement des fautes qui représente à son tour un élément fondamental de la SDF du SDC IEC 61850. La phase de vérification peut avoir lieu dans l'emplacement d'usine 'FAT', ou sur site lors de la mise en service du système 'SAT'. La FAT vise à simplifier les efforts de vérification réalisés en phase SAT pour réduire le temps de mise en service du système global de distribution d'énergie électrique [Haffar et al, 2010c].

Le paragraphe précédent a démontré la nécessité d'évaluer la fiabilité des échanges de communication au niveau de la phase de conception d'un SDC IEC 61850. Ce paragraphe va montrer la nécessité de l'utilisation de l'approche de Co-Simulation pour répondre aux aspects de vérification et de tests d'un SDC IEC 61850.

La liste des tests de communication qui doivent être effectués dans un tel système vont être abordés dans un premier temps, suivi par les aspects de vérification statique et dynamique faisant appel à une approche de Co-Simulation.

4.4.1 Programme de tests détaillé d'un SDC IEC 61850

Étant donné la complexité d'un SDC IEC 61850, plusieurs organismes sont à l'origine de l'accomplissement de ces tests. Par conséquent, il est vital d'avoir un programme détaillé pour réaliser cette phase et identifier la responsabilité de chaque organisme dans le programme de test.

Pour cela, il faut, dans un premier temps, avoir une image claire des différents types de test qui peuvent avoir lieu dans ce SDC.

Les tests de conformité sont des tests répandus dans le domaine de la communication. Le but essentiel de ces tests est de valider la conformité de l'implémentation du protocole de

communication dans les équipements par rapport à la spécification du standard [Udren et al, 2007b].

Chaque équipement IEC 61850 est livré avec deux documentations essentielles : le PICS '**P**rotocol **I**mplementation **C**onformance **S**tatement' et le MICS '**M**odel **I**mplementation **C**onformance **S**tatement' [INTERNATIONAL STANDARD IEC 61850-10]. Le PICS spécifie la capacité que possède l'équipement en terme de service de communication IEC 61850 implémenté dans ce dernier. Le MICS indique les fonctionnalités d'un équipement décrit sous forme d'objets conformes à la norme IEC 61850.

Plus spécifiquement, un test de conformité est généralement conduit par des organismes de certification type KEMA et UCAIUG [Tan et al, 2008] [Udren et al, 2007a] (figure 3.10). Ces organismes visent à vérifier la conformité du contenu des PICS et des MICS par rapport à l'aspect modélisation et aux services de communication définis dans la spécification des parties 7 et 8 et 9 de la norme [INTERNATIONAL STANDARD IEC 61850-7-x], [INTERNATIONAL STANDARD IEC 61850-8-1] et [INTERNATIONAL STANDARD IEC 61850-9-1].

Ces organismes ont recours à la partie 10 du standard IEC 61850 [INTERNATIONAL STANDARD IEC 61850-10]. Cette partie définit les directives afin d'établir un test de conformité efficace pour les équipements IEC 61850. L'équipement à tester sera ainsi nommé : DUT '**D**evice **U**nder **T**est'.

[Apostolov et al, 2004], [INTERNATIONAL STANDARD IEC 61850-10] et [Tan et al, 2008] listent les différents constituants de la plateforme qui permettent d'accomplir les tests de conformité. Le simulateur électrique et le composant de test IEC 61850 constituent les deux éléments essentiels de cette plateforme (figure 3.9).

Le simulateur électrique a pour but d'envoyer et/ou de recevoir des valeurs numériques et logiques vers et/ou depuis le DUT. Ces valeurs constituent une image des mesures électriques échangées entre les unités de protection et le procédé électrique. L'autre but essentiel de ce simulateur consiste à superviser les opérations effectuées par les unités de protection afin d'évaluer leurs performances (e.g. temps de déclenchement d'une sortie).

En fonction de la configuration implémentée dans le DUT, ce dernier procède à l'envoi des messages de communication évènementiels (i.e. Goose) suite à des changements de mesures électriques simulées. Ces messages sont reçus par le composant de test IEC 61850 qui procèdera à l'évaluation de leurs performances. Le composant de test permet aussi de générer des messages de communication (e.g. Goose) vers le DUT pour tester son comportement après réception de ces messages.

La figure 3.9 montre l'architecture typique du test de conformité.

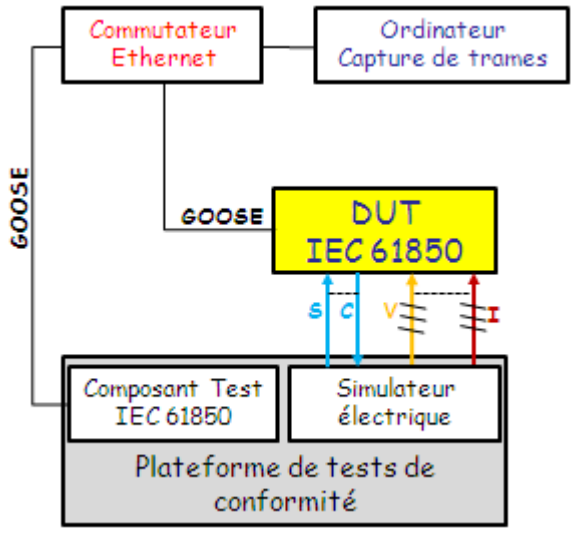


Figure 4.2 Architecture de test de conformité

Toutefois, il ne faut en aucun cas confondre les tests de conformité d'un DUT et les tests accomplis pendant les phases de FAT et SAT d'une architecture de communication IEC 61850. Un test de conformité réussi pour tous les équipements ne signifie pas que l'architecture de communication est adaptée aux exigences décrites dans sa spécification [Tan et al, 2008], il s'agit donc d'une condition nécessaire mais non suffisante.

Les organismes de certification proposent des directives pour effectuer des tests globaux sur l'architecture IEC 61850. Ces directives sont généralement traitées par des intégrateurs systèmes qui disposent d'outils avancés permettant donc le test d'une architecture IEC 61850 [Udren et al, 2007a].

Les intégrateurs systèmes sont considérés comme des tierces parties n'ayant aucune relation avec les constructeurs des unités de protection. La notion d'indépendance permet à ces intégrateurs de fournir une procédure de tests fiable, même en présence d'une architecture de communication IEC 61850 multi-constructeurs [Tan et al, 2008] [Udren et al, 2007a].

Pour avoir ce profil de test, les intégrateurs doivent, à leur tour, obéir à des conditions comme la présence d'experts dans le domaine IEC 61850 et la proposition d'un plan de tests bien défini basé sur des outils avancés ayant subi des validations par les organismes de certification [Tan et al, 2008] [Udren et al, 2007a] [Haffar et al, 2010b].

Les outils doivent finalement apporter des solutions permettant la réalisation efficace des tests pendant les phases de FAT pour donner un niveau de confiance élevé avant le déploiement final du projet sur site [Tan et al, 2008].

La figure 3.10 illustre les différents aspects des tests, la responsabilité des organismes dans le programme de test et les flux de directives échangés entre les organismes.

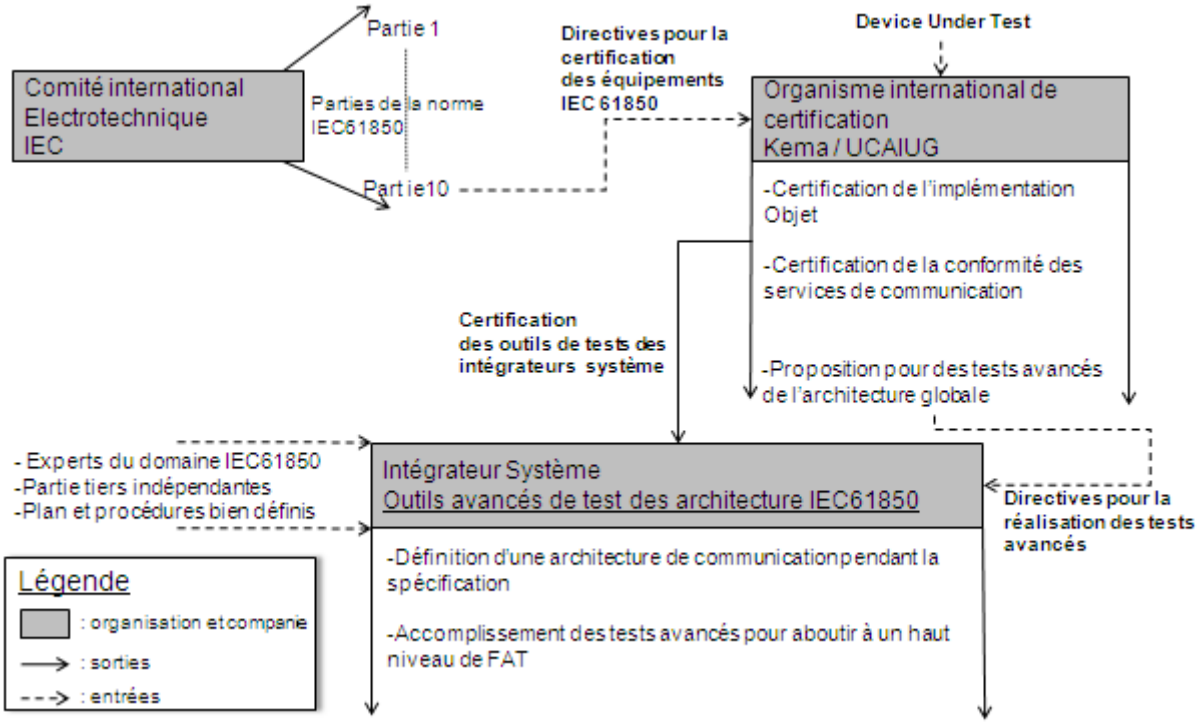


Figure 4.3 Organigramme détaillé des tests d'une architecture IEC 61850

4.4.2 Vérification statique d'un SDC IEC 61850

La vérification statique comme introduite dans {chapitre 2, paragraphe 2.3.7.2.1} consiste à effectuer les tests sur un système en mode non opérationnel.

Les industriels, et plus particulièrement les sociétés d'ingénierie, exigent la réalisation de cette procédure en emplacement usine (i.e. phase FAT) avant son lancement sur site (i.e. phase SAT) [Bruandet et al, 2010].

Cette exigence vise à minimiser le temps de mise en service des systèmes complexes. Le SDC IEC 61850 fait d'ailleurs partie de tels systèmes.

Les tests conventionnels appliqués à ce stade sont nommés tests verticaux 'Vertical Tests' [Bruandet et al, 2010] [Tan et al, 2009] (figure 3.11). Ces tests ont pour but de vérifier les flux de données échangés entre le niveau de protection et le niveau station (i.e. entre les unités de protection et les systèmes de supervision). Les signaux de télécommande, télésignalisation, télémesure, télé-contrôle et téléalarme sont les éléments essentiels étant à l'origine de ces flux de données. Ces signaux sont généralement issus du niveau station. Selon le signal reçu, la destination appartenant au niveau protection constituera sa réponse et l'envoi en retour au niveau station. Cet échange de flux est assuré par les services 'Pooling' (i.e. service de question/réponse).

Les fonctionnalités de communication avancées du standard IEC 61850 exigent l'ajout d'un ensemble de tests dans la catégorie des tests verticaux. Plus particulièrement, mis à part les services de communication 'Polling', d'autres types de services de communication sont

pourvus par la norme IEC 61850 pour permettre un dialogue vertical entre les unités de protection et les systèmes de supervision. Ainsi, on parle des services d'envoi de rapport de données. Ces services sont configurés pour être envoyés à l'initiative du serveur (i.e. unité de protection), soit selon un mode périodique soit selon un mode évènementiel (i.e. après détection d'un évènement électrique au niveau du serveur).

De plus, le fait que le standard IEC 61850 introduise des nouveaux services de communication, une nouvelle catégorie de tests intitulée 'Tests Horizontaux' [Bruandet et al, 2009] [Tan et al, 2009] sera ajoutée aux tests traditionnels 'Tests Verticaux' (figure 3.11). Cette catégorie vise à vérifier l'aspect publication et abonnement des signaux de communication GOOSE, entre les équipements de communication appartenant au niveau protection. Ces nouveaux types de tests doivent être impérativement menés lors de la phase de vérification statique d'un SDC IEC 61850.

Les deux catégories de tests rassemblées visent à vérifier la correspondance entre les paramètres de communication des fichiers CID {chapitre 3, paragraphe 3.2.5} et la description des échanges de données contenues dans la spécification du SDC IEC 61850.

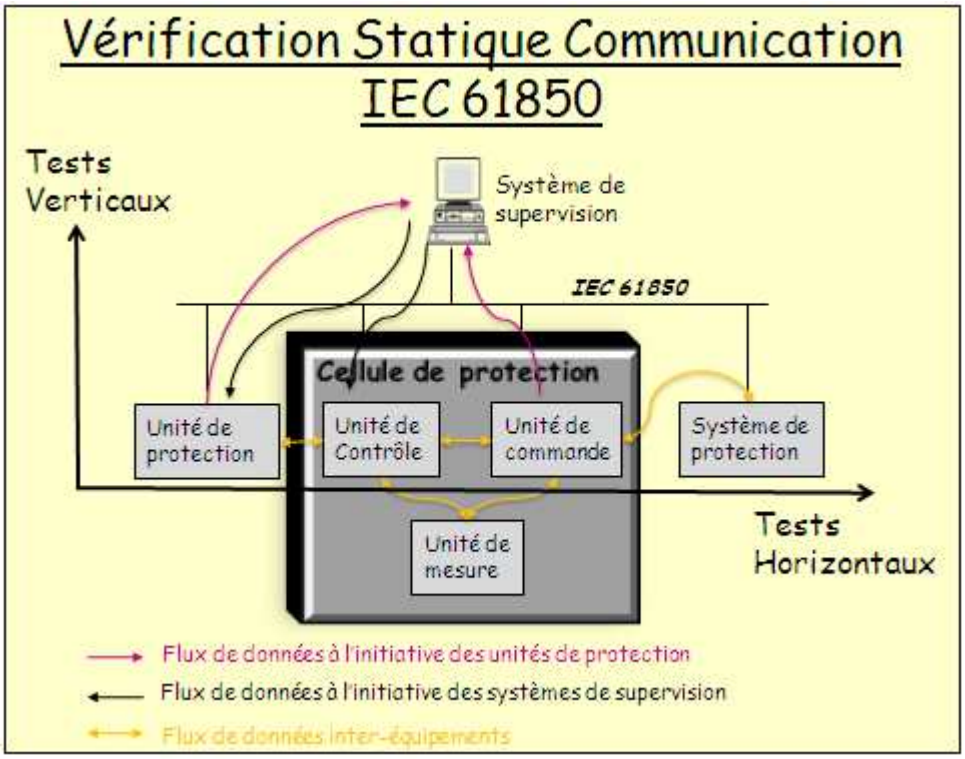


Figure 4.4 Catégories de tests de vérification statique d'un SDC IEC 61850

4.4.3 Vérification dynamique d'un SDC IEC 61850

La vérification dynamique comme introduite dans {chapitre 2, paragraphe 2.3.7.2.2} consiste à effectuer des tests sur un système en mode opérationnel. En se projetant sur le SDC IEC 61850, cette phase aura pour objectif de vérifier que les performances des signaux de communication Goose sont conformes aux critères de performances décrits dans les parties de la norme IEC 61850.

La catégorie de tests attribuée à cette vérification est intitulée 'Test Capacité Réseau' (figure 3.12). Le but de ce test est de vérifier la capacité qu'a le réseau à délivrer les performances des signaux Goose attendus en présence des vrais flux de données IEC 61850. Avec l'ajout de cette catégorie, l'aspect de vérification d'un SDC IEC 61850 (statique et dynamique) sera composé de trois axes de tests illustrés sur la figure 3.12.

La réalisation de cette vérification exige la présence de l'architecture de communication ainsi que les différents composants de communication constituant cette architecture. Cette exigence doit être respectée pendant la phase de FAT et pendant la phase de SAT et ce dans le but de réduire le temps de mise en service d'un nouveau SDEE.

Bien évidemment, l'approche de simulation ne peut pas être utilisée pour accomplir cette mission de vérification car les tests doivent être établis sur une plateforme physique composée d'équipements réels. De même, l'approche analytique ne peut être utilisée à cause de la complexité de cette architecture et de la nécessité de livrer des mesures de performances réelles.

La seule méthode qui pourra être appliquée à ce stade est l'approche expérimentale étant donné que les tests doivent être conduits sur l'architecture réelle. Néanmoins, le paragraphe suivant va montrer que cette approche fera face à des difficultés empêchant son utilisation au niveau de la phase de la FAT d'un SDC, d'où l'intérêt de l'approche de Co-Simulation.

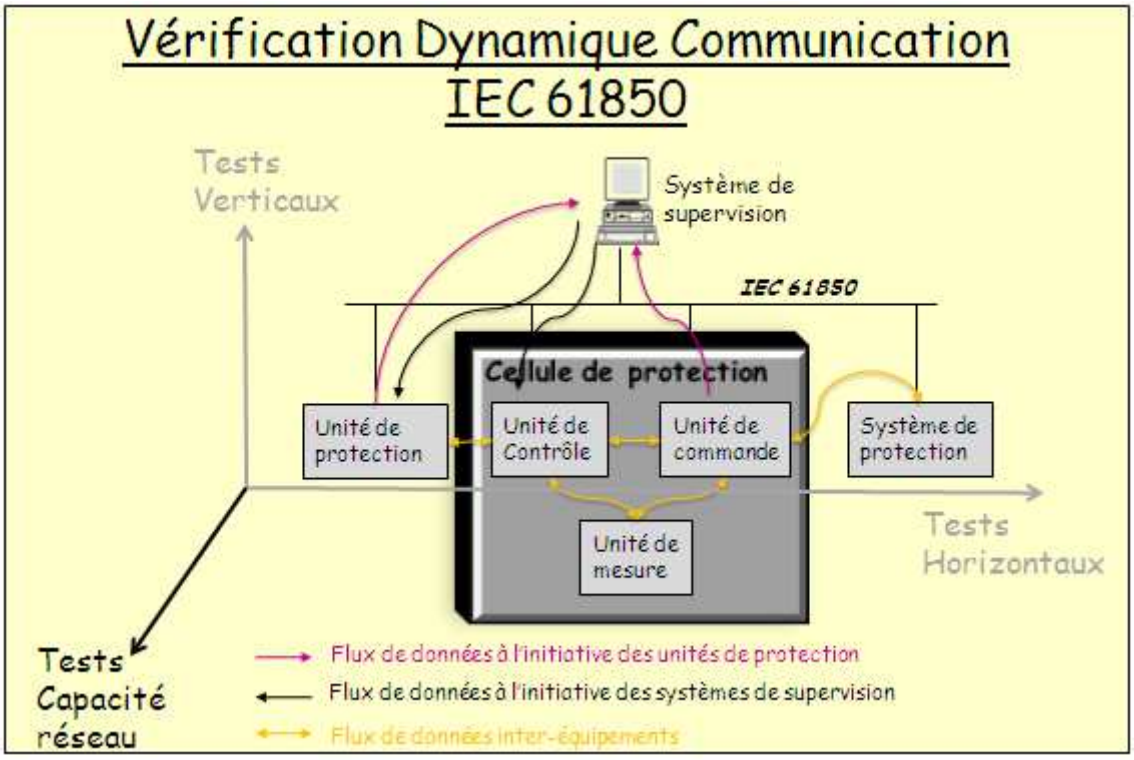


Figure 4.5 Catégorie de tests de vérification dynamique d'un SDC IEC 61850

4.4.4 Vérification en FAT et en SAT et approche par Co-Simulation

La **vérification statique** du SDC IEC 61850 peut être divisée en deux phases selon l'emplacement de son exécution.

Si l'exécution est faite à l'emplacement usine ou constructeur (i.e. phase de FAT), elle visera alors à vérifier la conformité des fichiers CID par rapport à la spécification de chaque équipement.

Pour mieux comprendre ce point, prenons l'exemple d'une simple spécification fonctionnelle introduisant une architecture de communication IEC 61850 hétérogène (i.e. multi-constructeur). Cette architecture bénéficie donc de la notion d'interopérabilité pourvue par le standard IEC 61850. L'architecture est supposée comme étant formée de trois unités de protection venant de trois constructeurs différents soient **'A'**, **'B'** et **'C'**.

Ces unités sont supposées être configurées de la manière suivante :

- l'unité 'A' envoie deux messages Goose dont l'un est destiné à l'unité 'B' et l'autre est destiné à l'unité 'C'. Le premier Goose intitulé 'GooseAB' est produit après un évènement électrique intitulé 'EvAB', le deuxième Goose 'GooseAC' est produit après un évènement électrique intitulé 'EvAC'.

- l'unité 'B' envoie un message Goose intitulé « GooseBA » à l'unité 'A' après l'apparition de l'évènement intitulé 'EvBA'

- l'unité 'C' envoie un message Goose intitulé GooseCA à l'unité 'A' après apparition de l'évènement intitulé 'EvCA'

Ainsi, les fichiers de configuration CID de chaque unité de protection doivent être conformes à cette spécification. Suite à l'implémentation de tous les fichiers CID dans les différentes unités de protection du SDEE (i.e. configuration complète de l'architecture de communication IEC 61850 du SDEE), la vérification statique devra avoir lieu.

Revenons à notre exemple et prenons le cas d'une vérification de l'unité de protection du constructeur 'A'. Les évènements 'EvAB' et 'EvAC' doivent être simulés sur cette unité de protection A pour vérifier la publication des messages 'GooseAB' et 'GooseAC'. D'autre part, pour finaliser la procédure de vérification statique de cette unité, il faut aussi tester la réception des messages 'GooseBA' et « GooseCA ». Pour cela, les évènements 'EvBA' et 'EvCA' doivent être simulés respectivement sur les unités de protection du constructeur B et du constructeur A.

La non-satisfaction de cette vérification (i.e. quatre tests dédiés à l'envoi/réception des quatre messages Goose) exige une reconfiguration des fichiers CID.

Pour le bon déroulement de cette procédure de vérification, une condition nécessaire doit être respectée. Cette condition exige la présence des unités de protection des constructeurs A, B et C au même emplacement lors de la phase FAT. Néanmoins, la notion d'interopérabilité empêche de satisfaire cette condition. En effet, lors de la phase de FAT, chaque unité de protection est placée à l'emplacement de son constructeur. La distance séparant les différents

constructeurs provoque l'absence de ces unités au même emplacement [Bruandet et al, 2010] [Rietmann et al, 2006] [Haffar et al, 2010a].

D'autre part, la vérification dynamique exige la présence de la totalité des équipements IEC 61850 y compris celle de l'architecture de communication choisie au niveau de la phase de conception. Généralement, le réseau de communication n'est installé physiquement que lors de la mise en service, ceci étant dû à la place requise pour l'installation de ce réseau. Par conséquent, la vérification dynamique du réseau de communication IEC 61850 (i.e. tests de performance des signaux de communication critiques Goose) ne peut pas prendre effet lors de la phase de FAT [Bruandet et al, 2010] [Haffar et al, 2010b].

La figure 3.13 illustre la problématique de vérification en phase de FAT.

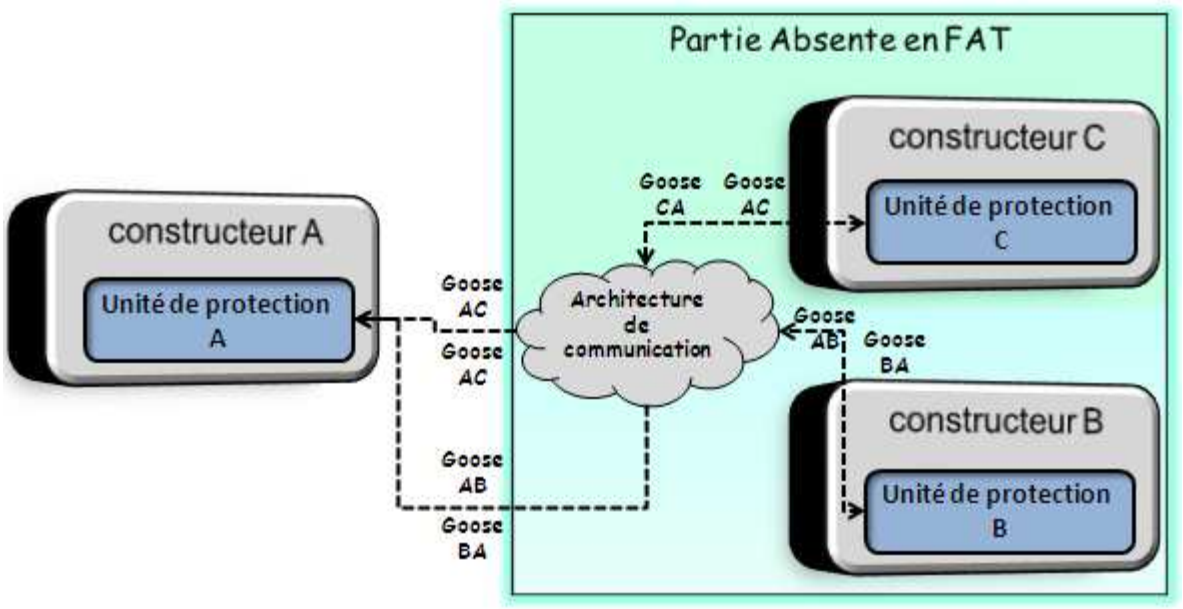


Figure 4.6 Problématique d'une FAT IEC 61850

Cet aspect est non négligeable pour rendre pertinente l'utilisation de l'approche de Co-Simulation qui permettra ainsi de résoudre les problématiques de vérification d'un SDC IEC68150.

La figure 3.14 permet d'illustrer l'intérêt de cette approche dans un environnement de communication IEC 61850.

Comme le montre cette figure, les modèles d'équipements sont générés en utilisant les vrais fichiers de configuration CID des équipements absents. Ainsi, ceci permet la réalisation de la vérification statique lors de la phase de FAT.

D'autre part, une architecture de communication simulée peut être construite dans l'environnement de Co-Simulation. Cette architecture visera à connecter les modèles des équipements absents aux unités de protection présentes lors de la phase de FAT, remplaçant ainsi l'architecture de communication réelle.

Les tests de performances des signaux Goose pourront ainsi se dérouler dans un environnement complètement dynamique combinant des composants appartenant à un monde réel avec des modèles existant dans l'environnement virtuel de Co-Simulation.

En utilisant cette approche, la vérification dynamique pourra avoir lieu dans la phase de FAT. Cette solution est un point fortement recommandé par la majorité des sociétés d'ingénierie électrique (Technip, Aramco, …) dans le but de réduire les problèmes au niveau de la mise en service ainsi que le temps de cette dernière {chapitre 2, paragraphe 2.3.7.2.2}.

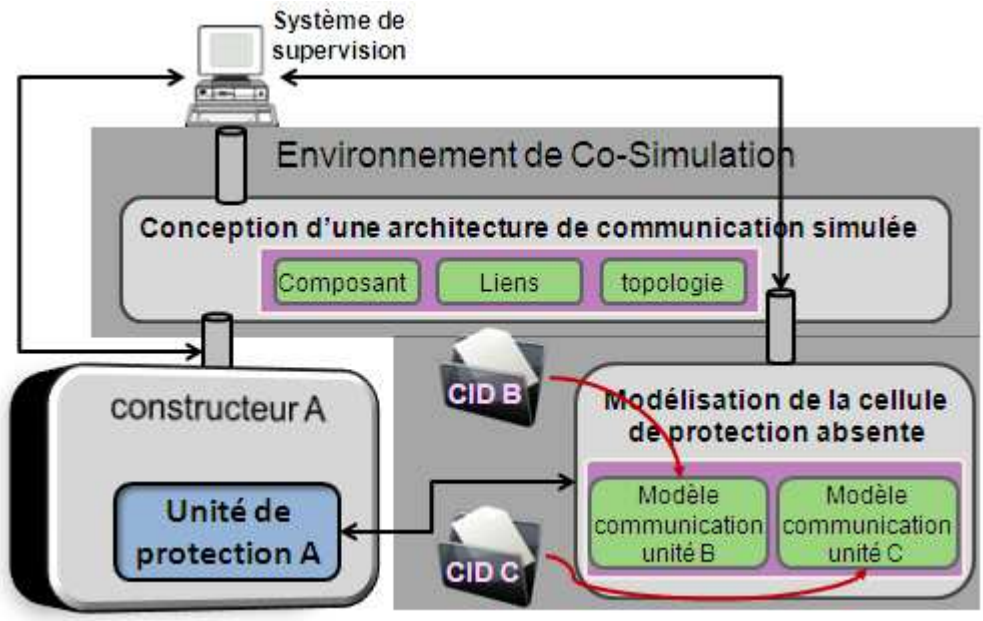


Figure 4.7 Approche de Co-Simulation pour la réalisation d'une FAT de haut niveau

4.5 Relation entre Co-Simulation et SDF d'un SDC IEC 61850

Les différentes études permettant l'évaluation de performance d'un SDC IEC 61850 {chapitre 3, paragraphe 3.4}, sont généralement réalisées au niveau des premières phases du développement d'un nouveau SDC IEC 61850 et ce, afin de choisir l'architecture de communication et les flux de données les mieux adaptés aux exigences fonctionnelles et normatives du SDC étudié. Nous avons expliqué dans {chapitre 3, paragraphe 3.5} que l'approche de simulation est la mieux adaptée pour l'estimation des performances des signaux de communication sécuritaire IEC 61850. Par conséquent, comme le montre la figure 3.15, la simulation (i.e. ou Co-Simulation ignorant la connexion avec le monde réel) constitue un élément essentiel du moyen de prévision des fautes d'un SDC IEC 61850. En particulier, cette simulation va donner des études prévisionnelles des défaillances temporelles pouvant affecter les messages de communication sécuritaire d'un SDC IEC 61850. Le résultat des études prévisionnelles permettent de justifier le choix de l'architecture d'un SDC IEC 61850 et les flux de données transitant dans cette architecture. [Thomas et al, 2010] contient un exemple de choix de l'architecture de communication en phase de conception. Les auteurs prouvent par l'intermédiaire des études prévisionnelles que l'architecture de communication choisie

dans leur étude est apte à faire passer les messages de communication sécuritaires IEC 61850 tout en respectant leurs exigences temporelles normatives.

Une fois l'architecture de communication choisie lors du moyen de prévision, les performances des échanges de communication doivent être revalidées pendant toutes les phases de tests et de vérification du SDC. Nous avons montré dans {chapitre 3, paragraphe 3.5} que les phases de vérifications statique et dynamique nécessitent la présence d'une approche de Co-Simulation, plus spécifiquement en phase de FAT du SDC IEC 61850. Par conséquent, la figure 3.15 montre que l'approche de Co-Simulation sera considérée comme l'un des éléments essentiels du moyen d'évitement des fautes d'un SDC IEC 61850.

Ce qui est primordial à retirer de la figure 3.15, est que l'approche de Co-Simulation permet de prendre en compte de la meilleure manière les deux moyens essentiels de la SDF du SDC IEC 61850 (i.e. prévision et évitement des fautes).

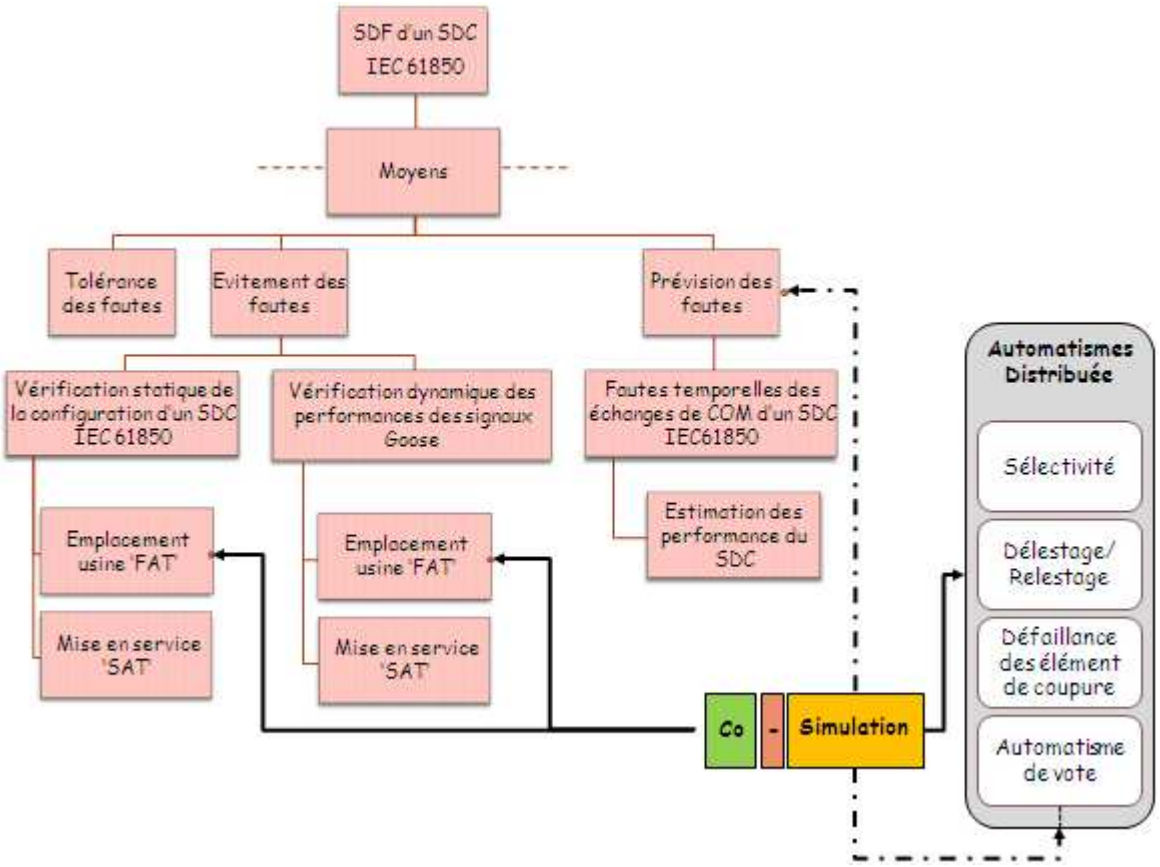


Figure 4.8 Co-Simulation Vs SDF d'un SDC IEC 61850

4.6 Conclusion

La partie de la SDF du SDC, prise en compte dans notre étude, retient l'évaluation des performances des échanges de communication IEC 61850.

Plusieurs approches (analytique, expérimentale, simulation, Co-simulation) existent pour réussir cette phase d'évaluation. La complexité du SDC IEC 61850 en termes d'échanges de plusieurs types de flux de données produisant des collisions aléatoires, rend difficile l'utilisation de l'approche analytique. Par ailleurs, la notion d'interopérabilité pourvue par le standard IEC 61850, induit des difficultés pour l'utilisation de l'approche expérimentale lors des phases de vérification du SDC.

Par conséquent, nous avons démontré tout au long de ce chapitre que l'approche par Co-Simulation est celle la plus appropriée pour l'évaluation de performances d'un SDC IEC 61850.

Cette approche sera un élément utile aussi bien pour le moyen d'évitement de fautes (i.e. vérification du SDC) que pour le moyen de prévision des fautes (i.e. conception de l'architecture de communication). En effet au niveau de la prévision, la Co-Simulation permet de générer des études de performance pendant la phase de conception pour identifier l'architecture de communication la mieux adaptée aux exigences normatives. Par ailleurs, cette approche est de même utilisée pendant la FAT pour accomplir les tests de la configuration globale du SDC (i.e. vérification statique) et la validation de la viabilité des messages sécuritaire (i.e. vérification dynamique).

Pour toutes ces raisons, les chapitres qui suivent vont expliquer la conception d'une plateforme de Co-Simulation et l'utilisation de cette plateforme pour des protocoles et des architectures de communication industriels.

Chapitre 5

Mise en œuvre d'une plateforme de Co-Simulation

5.1

5.1 Introduction

La Co-Simulation repose sur le développement d'une plateforme qui est basée sur un logiciel de simulation. Des modules supplémentaires sont interconnectés à ce logiciel, permettant ainsi de réaliser des simulations avancées. Cette interconnexion assure un échange de communication entre des composants réels et des modèles de simulation.

Ce chapitre va s'intéresser à la construction et au développement des deux modules essentiels de cette plateforme connus sous le nom de **'Hardware In The Loop' (HITL)** et **'Software In The Loop' (SITL)**.

Le module **HITL** assurera la communication entre les équipements réels existants hors de l'environnement de simulation et les modèles conçus dans le logiciel de simulation.

Le module **SITL** permet de créer des scénarios de simulation dynamique qui serviront à réduire le temps de celle-ci. Ce module assure également d'autres fonctionnalités avancées qui vont être abordées au cours de ce chapitre.

Plusieurs composants logiciels, qui sont interconnectés, constituent ces modules. Certains composants sont disponibles dans la bibliothèque du logiciel de simulation ou dans celle du système d'exploitation et d'autres ont été développés dans le cadre de notre étude.

Ce chapitre abordera aussi :

- les difficultés de programmation rencontrées lors du développement et du test de certains composants des modules HITL et SITL,
- l'interconnexion des composants HITL et SITL avec le logiciel de simulation.

5.2 Choix de l'environnement de simulation

La modélisation et la simulation des architectures de communication nécessitent un simulateur de réseau. Celui-ci devra prendre en compte toutes les spécificités du réseau, notamment au niveau des composants et des paramètres constituant ce dernier.

Ils existent de nombreux logiciels sur le marché permettant de réaliser ce genre de simulation (OpNet (**OP**timized **N**etwork **E**ngineering **T**ool, TrueTime, NS2 **N**etwork **S**imulator **2**, etc.).

TrueTime est un module de simulation réseau proposé par Matlab. Le but essentiel de ce module est de permettre aux contrôleurs développés dans Matlab d'échanger des informations et commandes avec des capteurs et actionneurs à travers un réseau de communication simulé. Toutefois, cet outil est dédié à de simples applications de simulation [Chitnis et al, 2007] du fait qu'il ne dispose pas d'une riche librairie de fonctions dédiées au développement des protocoles de communication et des échanges de paquets. En plus, à la différence des logiciels proprement dédiés pour la simulation des réseaux de communication, cet outil ne propose pas une riche librairie de modèles de simulation. Pour ces raisons, cet outil n'a pas été choisi dans notre étude.

NS2 est un logiciel de simulation libre de droit qui offre les mêmes avantages qu'OpNet en termes de fonctionnalités et de librairies de fonctions et de modèles. Son aspect libre de droit permet d'une part une utilisation gratuite de cet outil mais d'autre part l'absence à une assistance technique qui demeure indispensable pour un développement de recherche industrielle.

Des études comparatives entre les deux logiciels ont été introduites dans [Lucio et al, 2004], [Garrido et al, 2005]. Les auteurs proposent des simulations en utilisant les deux logiciels et comparent les résultats à des expérimentations réelles. Leurs études montrent que les deux simulateurs arrivent bien à fournir des résultats proches de ceux obtenus avec les expérimentations. Toutefois ils montrent que l'avantage qu'a OpNet sur NS2 réside au niveau de la construction des modèles de simulation. Cette construction est basée sur une programmation graphique en OpNet en utilisant des graphes d'états et passant vers une programmation textuelle en NS2.

Vu la complexité et le nombre de modèles qu'on est mené à développer dans notre étude, notre choix s'est basé sur le logiciel OpNet et ce pour bénéficier d'autres intérêts listés dans [Guo et al, 2007] :

- Produit industriel, ce qui facilite la tâche de dépannage pour une thèse en ayant une finalité de développement
- Mise à disposition d'une riche librairie d'équipements réseau standard (Commutateur, Routeur, Liens etc)
- Mise à disposition d'une riche librairie de fonctions informatiques pour faciliter le développement des modèles

5.3 Modélisation avec le logiciel OpNet Modeler

Chaque modèle de simulation développé dans le logiciel OPNET Modeler est intitulé nœud ou **'Node'** et possède un modèle de nœud intitulé **'Node Model'**.

Le modèle de nœud est décomposé en plusieurs modules interconnectés entre eux à travers des liens intitulés **'Packet Stream'**. Ces modules forment une image des couches de communication OSI de l'équipement réel modélisé.

Chaque module sert à modéliser un aspect interne du comportement du modèle de simulation (i.e. nœud OpNet). En prenant l'exemple du module applicatif, il sert à la construction des paquets conformément au protocole de communication existant dans la couche applicative. Le module TCP, quand à lui, permet d'assurer un mécanisme fiable de la transmission des données entre les nœuds OpNet.

Pour atteindre ses objectifs, chaque module OpNet doit être programmé par l'intermédiaire d'un éditeur intitulé **'Process Editor'**. Cet éditeur permet d'attribuer à chaque module OpNet un modèle intitulé **'Process Model'**.

La programmation d'un **'Process Model'** est effectué à partir d'un graphe d'état (**F**inite **S**tate **M**achine-**FSM**). Cette technologie est basée sur des étapes et des transitions permettant de définir la progression du **'Process'** du module en question.

A leurs tours, chaque étape contient du code C/C++ constituant le programme de l'étape. Les étapes utilisent les librairies de fonctions fournies par le logiciel OpNet dans le but de faciliter la tâche de modélisation des protocoles de communication. Ces librairies sont programmées à leur tour avec le langage C/C++. Chaque étape possède un code qui s'exécute à l'entrée de l'étape **(Enter Executive)** et un code qui s'exécute à la sortie de l'étape **(Exit Executive)**.

Pour donner plus de souplesse à la technologie de graphes utilisé dans le 'Process Model', OpNet fournit deux catégories d'étapes et deux catégories de transitions. Une étape peut être considérée comme forcée ou non forcée tandis qu'une transition peut être considérée comme

conditionnée ou non conditionnée. Dans le cas d'une transition conditionnée, la condition associée à cette transition doit être programmée en langage C/C++.

Une étape non forcée, représentée en gris foncé sur la figure 4.1, permet un blocage entre l'entrée et la sortie de cette étape. Lorsque le 'Process Model' exécute cette catégorie d'étape, il se bloque à la fin du code d'entrée de l'étape. L'exécution du code de sortie de l'étape est conditionnée par la réception d'une interruption permettant de débloquer le **'Process'**.

Une étape forcée, représentée en gris clair sur la figure 4.1, ne permet pas le blocage du Process entre le code d'entrée et le code de sortie. Par conséquent, à l'exécution de cette catégorie, le code de sortie sera automatiquement exécuté après l'exécution du code d'entrée.

Les transitions sont représentées par des flèches liant la sortie d'une étape vers l'entrée d'une autre étape. Une transition conditionnée est représentée par une flèche pointillée tandis qu'une transition non conditionnée est représentée par une flèche continue.

Plusieurs transitions peuvent être issues d'une même étape. Dans ce cas, les conditions associées à ces transitions doivent être complémentaires pour ne pas générer une erreur d'exécution de la simulation.

Cette approche de programmation du Process Model donne une grande facilité pour l'implémentation des algorithmes, des applications et des protocoles de communications.

La figure 4.1 illustre les différents éditeurs de programmation du logiciel OpNet ainsi qu'un exemple symbolique du contenu de chaque éditeur.

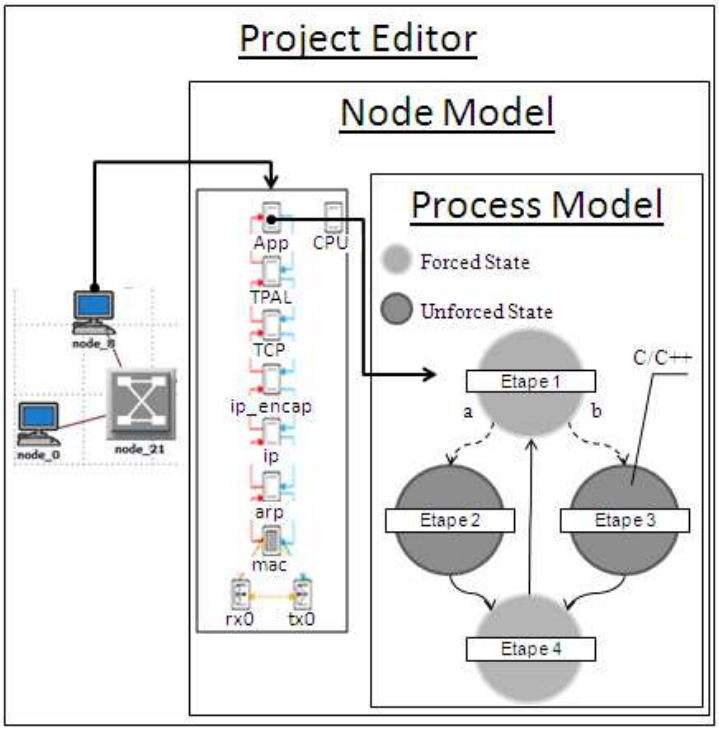


Figure 5.1 Environnement de programmation du logiciel OpNet Modeler

5.4 Vue structurelle de la plateforme de Co Simulation

La figure 4.2 représente la vue structurelle de la plateforme de Co-Simulation qui contient les logiciels de simulation et de modélisation mais aussi les modules HITL et SITL. Tous ces modules sont installés dans une seule machine appelée 'Machine de Co-Simulation'.

Cette machine englobe:

- Le Logiciel de simulation et de modélisation
- Un module HITL
- Un module SITL

Les modèles de communication ainsi que l'implémentation des protocoles de communication industriels sont réalisés dans le logiciel de simulation.

Une interface de communication virtuelle paramétrable est associée à chaque modèle de communication. Cela leur permet de dialoguer entre eux ou avec des équipements réels.

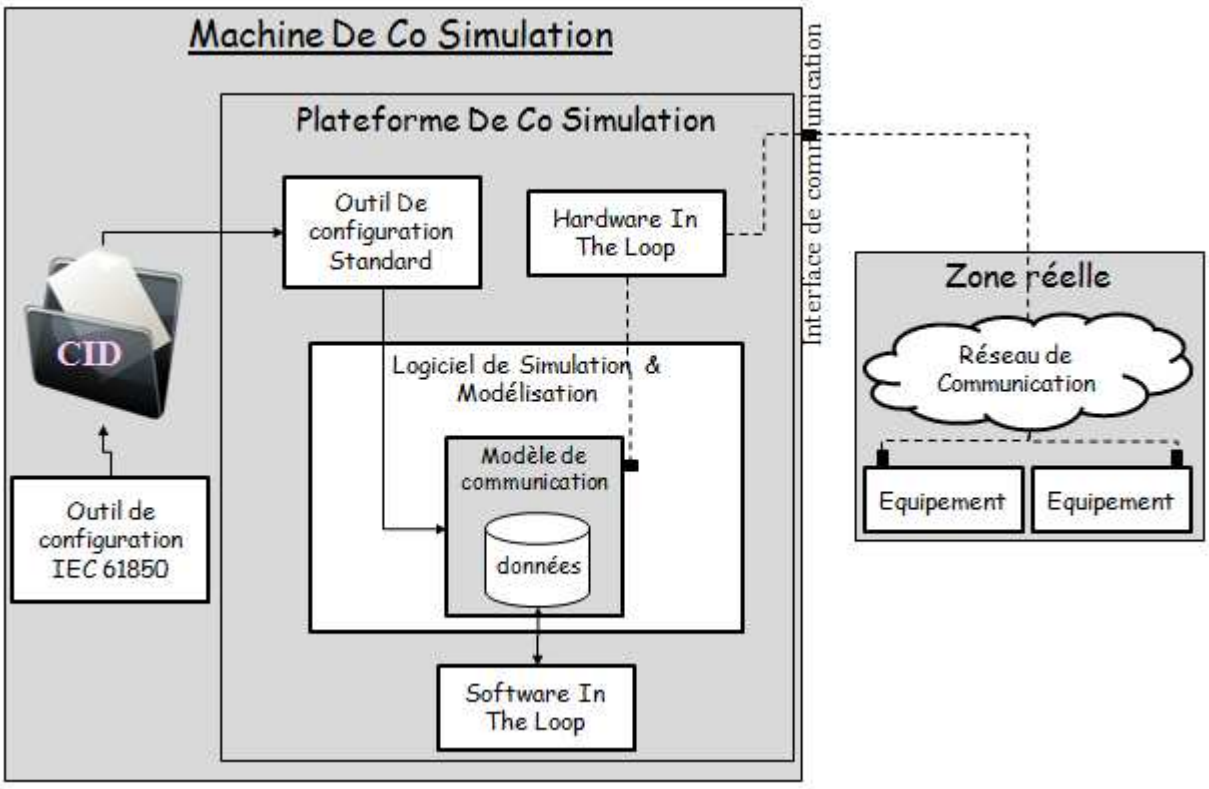


Figure 5.2 Vue structurelle de la plateforme de Co Simulation

L'HITL est un module qui assure la communication entre les modèles de communication conçus dans le logiciel de simulation et les équipements physiques qui appartiennent à une zone réelle. L'interface de communication de la machine de Co-Simulation permet à ces deux entités de communiquer.

Le SITL est un module qui donne, à un outil externe au logiciel de simulation, un accès direct aux variables et au comportement des modèles de simulation. Ce module permet un accès à la simulation en mode 'exécution'. Les détails et l'intérêt de l'implémentation de ce module sont exposés dans le paragraphe 4.6.

5.5 Module HITL d'OpNet

Le module HITL est un élément optionnel du logiciel de simulation OpNet Modeler. L'utilisation de ce module permet d'assurer les fonctionnalités suivantes :

- Construction d'une architecture mixte comprenant des modèles et des équipements physiques. Les modèles comblent l'absence de certains équipements physiques lors de la phase d'évaluation de la performance du système.
- Validation des protocoles de communication qui sont implémentés dans les modèles d'équipement.
- Test de l'interopérabilité des équipements réels.

5.5.1 Architecture de base établie via le HITL

Plusieurs architectures peuvent être mises en place par l'utilisation du module de simulation HITL. Chacune de celles-ci peut résoudre une problématique bien spécifique dans le monde de la modélisation et de la simulation de réseaux.

Ce module produit différentes architectures de base dont : **RS** '**R**éel **S**imulé', **RSR** '**R**éel **S**imulé **R**éel' et **SRS** '**S**imulé **R**éel **S**imulé'. Ces architectures peuvent aussi être combinées pour en construire d'autres, plus complexes.

La première architecture RS, illustrée par la figure 4.3, est caractérisée par un échange direct entre un réseau réel et un réseau simulé. Elle permet d'assurer les fonctionnalités suivantes :

- Les tests d'interopérabilité des équipements réels. En effet, un modèle de communication est considéré comme un équipement indépendant de tous les constructeurs du marché. Ainsi, le fait de valider des échanges de communication entre ce modèle et un équipement réel permet de vérifier l'interopérabilité de ce dernier.
- Validation de la modélisation des protocoles de communication. Cette méthode consiste à connecter le modèle en question à un système certifié conforme à ce protocole. Ceci permet de vérifier les échanges entre les deux entités. Dans ce cas, l'architecture RS inclura uniquement le système réel certifié et le modèle à tester.

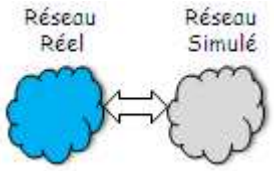


Figure 5.3 Architecture de Co-Simulation RS

La deuxième architecture, RSR, est illustrée par la figure 4.4. Utilisée dans les systèmes commandés en réseau, elle pourvoit à l'échange de communication entre deux réseaux réels par l'intermédiaire d'un réseau simulé. Ces systèmes exigent une évaluation de la qualité de contrôle des équipements réels en fonction de la qualité de service d'un réseau de communication simulé.

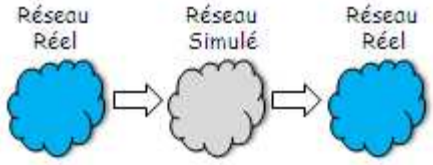


Figure 5.4 Architecture de Co-Simulation RSR

Illustrée par la figure 4.5, SRS, qui est la dernière architecture, permet à deux zones simulées de se connecter par l'intermédiaire d'un réseau réel.

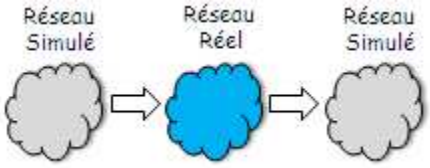


Figure 5.5 Architecture de Co-Simulation SRS

5.5.2 Objets HITL et enchaînement de paquets

Lorsque l'interface de communication de la machine de Co Simulation reçoit un paquet, plusieurs éléments HITL, indiqués par la figure 4.6, pourvoient à son routage vers le modèle de simulation.

Le 'composant HITL', qui est le premier élément, permet la récupération du paquet de l'interface de communication de la machine de Co-Simulation, puis sa transmission à la 'passerelle HITL' si la simulation est en mode exécution.

La 'passerelle HITL' constituant le deuxième élément du module HITL, permet la conversion du paquet depuis le format réel vers le format de simulation. Elle l'envoie enfin à destination par l'intermédiaire du 'lien HITL'.

Le 'lien HITL', le troisième élément du module HITL, assure la connexion entre la passerelle et le modèle de destination (ex. commutateur, répéteur ou simple modèle Ethernet).

Ces trois éléments sont livrés par OpNet Modeler et existent dans la bibliothèque des modèles de ce logiciel.

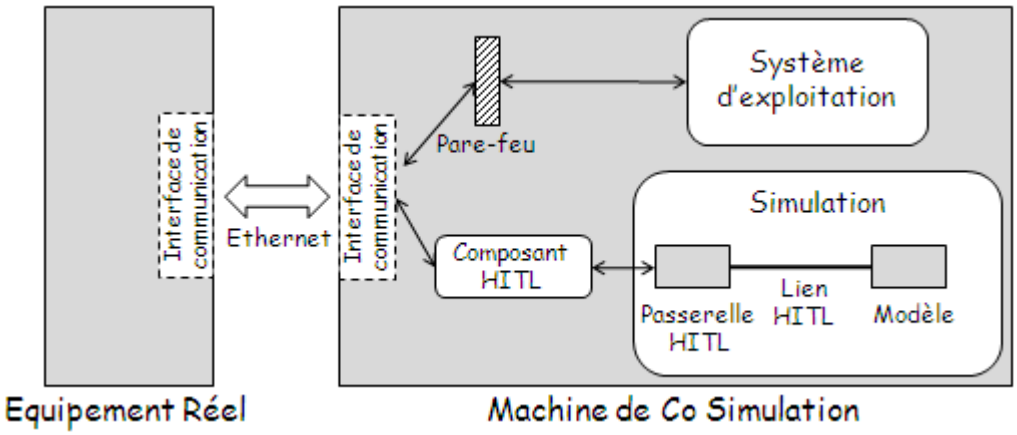


Figure 5.6 Enchaînement des paquets entre les objets HITL

5.5.3 Cycle de conversion de la passerelle HITL

La figure 4.7 illustre les différentes étapes du cycle de conversion de la passerelle HITL.

La première étape, appelée **'Mappage',** permet de mapper l'interface de communication de la machine de Co-Simulation à l'interface de communication du modèle de simulation.

La deuxième étape, appelée **'Filtrage'**, applique un filtre qui réduit le nombre de paquets qui transitent via la simulation. Le filtrage est établi par l'intermédiaire d'une configuration spécifique d'un des attributs de la passerelle HITL. La condition de filtrage doit respecter la syntaxe de l'utilitaire 'WinPcap de Windows', puisque notre plateforme de développement est une plateforme Windows.

Lorsqu'un paquet franchit la condition de filtrage, il entre dans la troisième étape du cycle de conversion, la **'Création'**. Celle-ci renferme la création du squelette du paquet (i.e. son format), à convertir en fonction du paquet reçu. Ces paquets simulés sont de deux types :

a- Type formaté : Ce type est construit en utilisant l'éditeur de construction de paquet d'OpNet Modeler {Annexe B},

b- Type non formaté : Ce type est construit par l'intermédiaire d'une allocation mémoire et d'une juxtaposition des champs de la mémoire allouée {Annexe B}.

Si le paquet reçu est simulé, la procédure de création a recours à un emplacement mémoire spécifique à partir duquel les champs du paquet réel doivent être remplis.

La phase de création est suivie de la phase de **'translation et de mise à jour'**. Celle-ci a essentiellement pour but de copier les valeurs des champs du paquet original dans les champs vides du paquet de conversion qui a été créé au préalable. Cette conversion sera effectuée par des fonctions spéciales permettant la lecture des valeurs du paquet original et l'écriture de ces valeurs dans les champs de paquet à convertir {Annexe A}. La translation est normalement établie par des tranches de paquet ; un paquet est découpé en plusieurs tranches qui appartiennent chacune à une couche protocolaire bien spécifique. La translation de la couche applicative fera l'objet d'un traitement spécial exposé dans les paragraphes suivants.

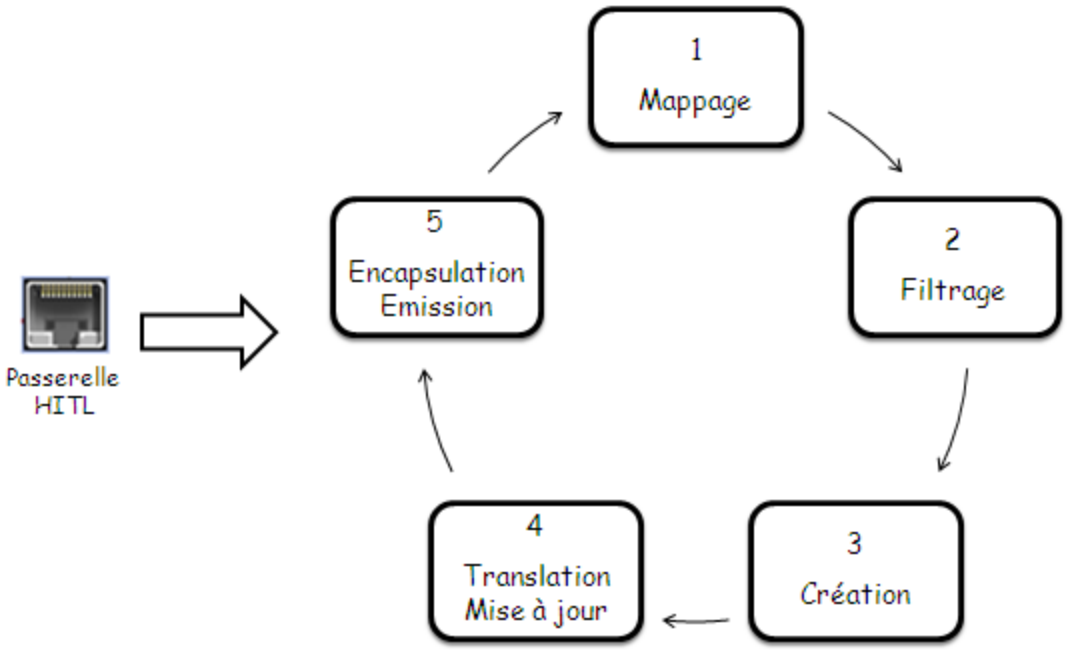


Figure 5.7 Cycle de conversion de la passerelle HITL

L'étape 5 du cycle de conversion sera lancée après la procédure de translation. Chaque paquet translaté sera encapsulé avec ses prédécesseurs, pour constituer le paquet final de conversion.

Si aucune des tranches du paquet final n'a subi d'échec tout au long des étapes 1 à 5, la conversion sera réputée réussie et le paquet final converti sera transmis à la zone opposée à celle du paquet d'origine. Dans le cas contraire, le paquet sera rejeté dans la passerelle HITL.

5.5.4 Attributs associés à la passerelle HITL

La passerelle HITL d'OpNet possède plusieurs paramètres configurables qu'on appelle des attributs. Ces derniers, illustrés par la figure 4.8, définissent le mode de fonctionnement de cette passerelle. Une partie de ces attributs doit obligatoirement être configurée tandis que l'autre est optionnelle.

Les flèches noires de cette figure indiquent les attributs à configurer obligatoirement, dont la syntaxe de filtrage et le choix de l'interface de communication de la machine de Co-Simulation notamment, des paramètres devant impérativement être spécifiés par l'utilisateur.

Par ailleurs, l'utilisateur doit également définir trois autres attributs liés aux fonctions de translation de la couche applicative. Ces derniers opéreront une translation totale de paquets (i.e. conversion réussie). D'ailleurs, ils feront l'objet d'explication à la suite de ce chapitre.

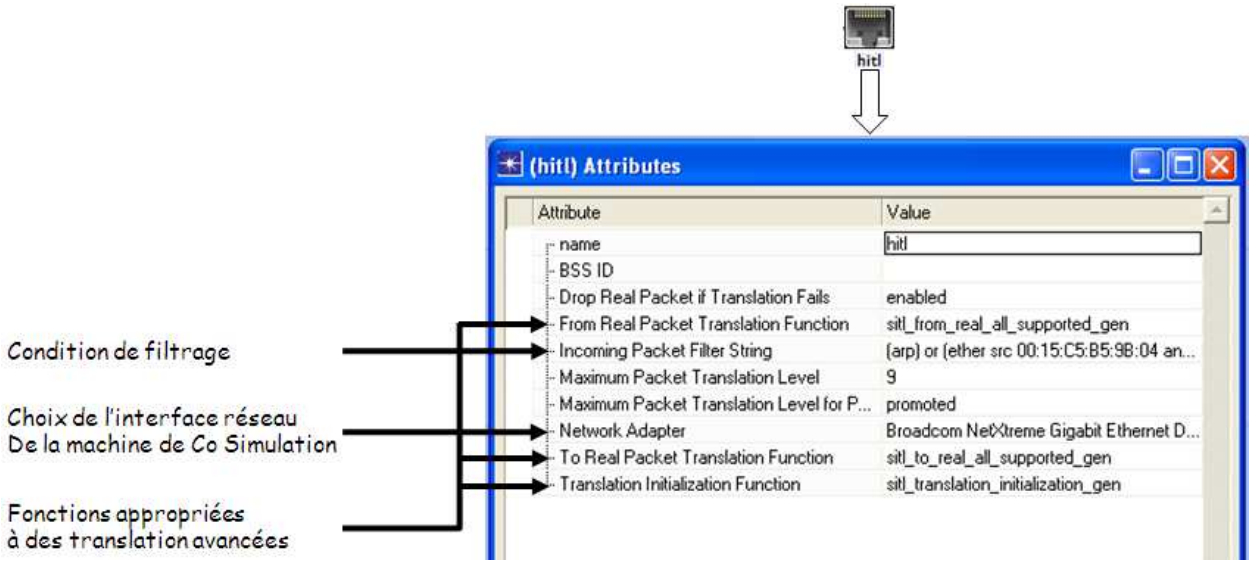


Figure 5.8 Attributs de la passerelle HITL

5.5.5 La procédure de translation des paquets

La quatrième étape du cycle de conversion comportera la procédure de translation. Cette dernière, garantie par la passerelle HITL, consiste à effectuer la translation des paquets dans les deux sens (i.e. les paquets simulés sont convertis en réel et réciproquement).

Le principe de cette procédure est le suivant:

Lorsque la passerelle reçoit un paquet (virtuel ou réel), elle procède à sa conversion en déterminant le format de chaque sous-paquet et en appliquant les fonctions de conversion appropriées. La passerelle intègre des fonctions de translation standard, qui sont appliquées à des protocoles de communication spécifiques appartenant à des couches réseau bien définies.

La détermination du format de paquet est établie en utilisant des fonctions de test de signature appliquées à des champs particuliers de ce paquet.

Si le test de signature donne comme résultat que le paquet reçu fait partie des formats reconnus par la passerelle, une fonction de translation associée au sous-paquet sera invoquée pour effectuer sa translation. Dans le cas contraire, deux cas peuvent se présenter, soit une fonction utilisateur est développée permettant de prendre en charge la translation de la partie non reconnue du paquet, soit le paquet sera rejeté par la passerelle HITL.

Le paragraphe 4.5.5.1 présente les différents formats de paquets supportés, en natif, par la passerelle HITL.

5.5.5.1 Format de paquets supportés par HITL

HITL permet de convertir un nombre limité de paquets. Ces derniers possèdent un format bien défini et appartiennent à des couches de communication bien particulières.

Le tableau 4.1 répertorie les fonctions de translation dans le sens réel/simulé (RS) des différents formats de paquet supportés par la passerelle HITL. Il existe les mêmes fonctions

en substituant le ‘from' par ‘to’ dans le tableau pour la prise en compte des translations des paquets venant du monde simulé et transitant vers le monde réel (SR).

Tableau 5.1 Liste des protocoles supportés par la passerelle dans le sens (RS)

Couche/protocole	Format de simulation	Fonction de translation
Liaison/Ethernet Liaison/ARP	ethernet_v2 arp_v2	op_pk_sitl_translate_ from_real_ethernet () op_pk_sitl_translate_ from _real_arp ()
Réseau/IPV4 Réseau/ICMP Réseau/RIP	ip_dgram_v4 ip_icmp_echo rip_message2	op_pk_sitl_translate_ from _real_ipv4_ip () op_pk_sitl_translate_ from _real_ipv4_icmp () op_pk_sitl_translate_ from _real_ipv4_rip ()
Transport/TCP Transport/UDP	tcp_seg_v2 udp_dgram_v2	op_pk_sitl_translate_ from _real_ipv4_tcp () op_pk_sitl_translate_ from _real_ipv4_udp ()
Union de toutes les couches et protocoles	Tous les formats mentionnés ci-dessus	op_pk_sitl_ from_real_all_supported ()
Application	Aucun format supporté	Pas de fonction de translation

Le fait que la passerelle HITL soit programmée pour effectuer la translation des couches basses du protocole TCP/IP (liaison, réseau, transport) est d’une importance notoire. La passerelle HITL prend automatiquement en charge les formats de paquets associés aux couches Ethernet, IP, TCP et UDP.

Toutefois, la passerelle HITL ne prend pas en charge la translation des paquets appartenant à la couche applicative. Ceci contribue à un échec de communication entre les équipements réels et les modèles de simulation.

Par conséquent, l’utilisation réussie du module HITL exige un développement des fonctions de translation utilisateur, permettant ainsi la translation des paquets applicatifs. Une translation complète des paquets sera donc assurée. Nous avons programmé dans notre étude une librairie de fonctions permettant d’assurer cette tâche de translation pour n’importe quel protocole de communication implémenté dans la couche applicative des modèles de simulation {Annexe A}. On va montrer dans les sous-paragraphes suivant la création d’une librairie de fonction dont les fonctions seront injectés dans les attributs de la passerelle HITL afin d’assurer cette translation totale des paquets dans les deux sens de communication.

La création de cette librairie a été un développement important de notre travail de recherche qui a nécessité le passage par plusieurs versions pour arriver à la librairie finale permettant le bon fonctionnement de la passerelle HITL pour des applications simulées et réelles fonctionnant au delà des couches TCP/IP.

5.5.5.2 Première version de la librairie HITL

Dans cette première version, deux blocs de fonctions informatiques C ont été développés pour assurer la translation des données applicatives dans les deux sens.

Chaque procédure de translation a recours à un bloc de contrôle HITL contenant toutes les informations utiles pour accomplir une translation. Les différents champs compris dans ce bloc de contrôle sont listés dans l'annexe {Annexe A}.

La figure 4.9 illustre les étapes relatives à la procédure de translation des données applicatives dans le sens simulation réel. Les détails des fonctions informatiques développées dans cette version sont montrés dans l'annexe {Annexe A}.

La première étape consiste à extraire le paquet de simulation arrivant à la passerelle HITL ; ce paquet est localisé dans le bloc de contrôle de la passerelle.

L'évaluation de la taille du paquet de simulation reçu constitue la deuxième étape. Elle a pour but de faciliter la programmation des fonctions de translation.

La troisième étape consiste en l'extraction des données contenues dans le paquet de simulation. Ces données sont rangées dans une structure C.

La dernière étape inclut la transcription des valeurs de champs de simulation vers les champs constituant le paquet réel. Les champs du paquet réel appartiennent à une zone mémoire spécifique définie dans le bloc de contrôle HITL.

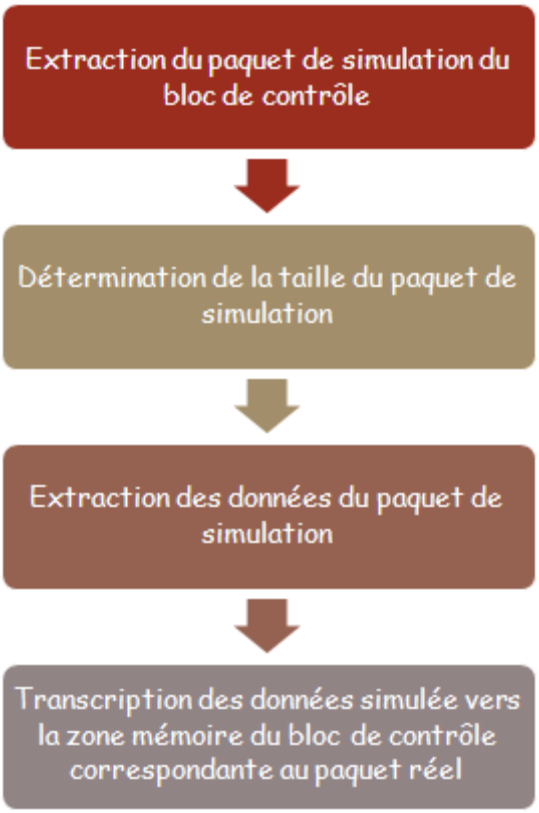


Figure 5.9 Etape de translation de paquet dans le sens simulation réel

La deuxième procédure montrée sur la figure 4.10 illustre les différentes étapes pour la translation dans le sens réel simulation. Les détails de programmation de cette procédure sont également illustrés dans l'annexe {Annexe A}.

Les deux premières étapes visent à extraire les données correspondant à la couche application du paquet réel reçu et à les copier dans une zone de mémoire interne de simulation.

Pour cela, nous avons d'abord alloué une mémoire de taille égale à celle de la couche applicative du paquet réel, puis nous l'avons rempli par des données contenues dans le bloc de contrôle. Ces données correspondent à celles de la couche applicative du paquet reçu.

Ensuite, nous avons construit le paquet de simulation. Cette étape est divisée en deux parties. En premier lieu, la création d'un paquet de simulation vide et en second lieu, le remplissage de ce paquet par les valeurs de la zone mémoire construite dans les premières étapes.

La dernière étape est l'injection du paquet de simulation, construit au cours des étapes précédentes, dans son emplacement dans le bloc de contrôle. Cette dernière étape permet à la passerelle de transférer le paquet de simulation vers le modèle (i.e. nœud) de simulation via le lien HITL.

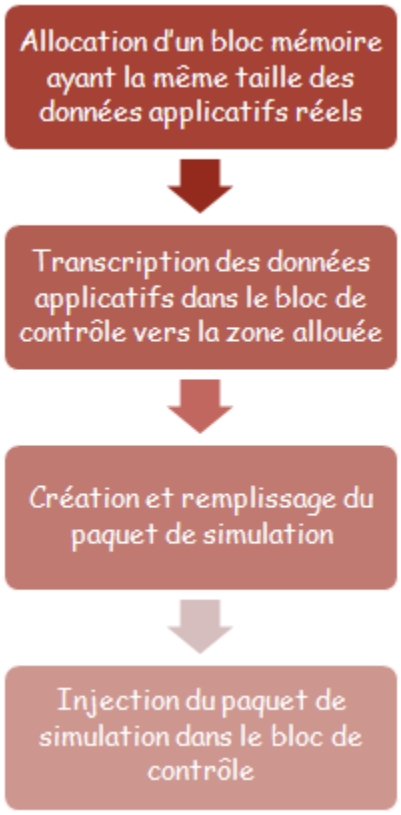


Figure 5.10 Etape de translation de paquet dans le sens réel simulation

5.5.5.3 Résultat d'implémentation de la première version de la librairie HITL

Cette première version de la librairie a été implémentée dans le logiciel de simulation OpNet Modeler en respectant le schéma d'implémentation décrit dans l'annexe {Annexe A}.

Le résultat de l'implémentation de cette première version de la librairie est illustré sur la figure 4.11. Ce résultat résume que cette version n'a pas abouti à un résultat satisfaisant dans les deux sens d'envoi.

En effet, un paquet envoyé depuis un équipement réel n'est pas reçu par la couche applicative à qui il est destiné (i.e. modèle de simulation). Après un débogage informatique effectué en utilisant la console de débogage OpNet ODB (OpNet Debugger), on a constaté une conversion totale du paquet envoyé depuis le monde réel sans qu'il soit rejeté par la passerelle HITL (figure 4.11). Néanmoins, lors de la décapsulation du paquet de simulation converti au niveau des couches réseau du nœud de simulation et en particulier, au niveau de la couche transport, le segment de paquet associé à cette couche a signalé la présence d'une erreur. Par conséquent, la décapsulation finale qui mène à l'avancement de la dernière partie du paquet vers la couche applicative du nœud de simulation, n'a pas pu prendre effet (figure 4.11). Cela a abouti à la non-réception du paquet applicatif par le modèle de simulation.

Dans l'autre sens, un paquet envoyé par un modèle de simulation a eu le même cheminement mais n'arrive pas à destination dans le monde réel. Un débogage informatique, réalisé en utilisant l'utilitaire Wireshark, a confirmé que le paquet envoyé par le nœud de simulation n'a pas été transmis via l'interface de communication de la machine de Co-Simulation (figure 4.11). Ceci a été confirmé par le débogage, en utilisant la console OpNet Debugger, qui a indiqué un échec dans le cycle de conversion du paquet qui a donc été rejeté par la passerelle HITL.

Ces deux points nous ont amené à modifier la librairie pour passer à une nouvelle version permettant de résoudre ces problèmes rencontrés.

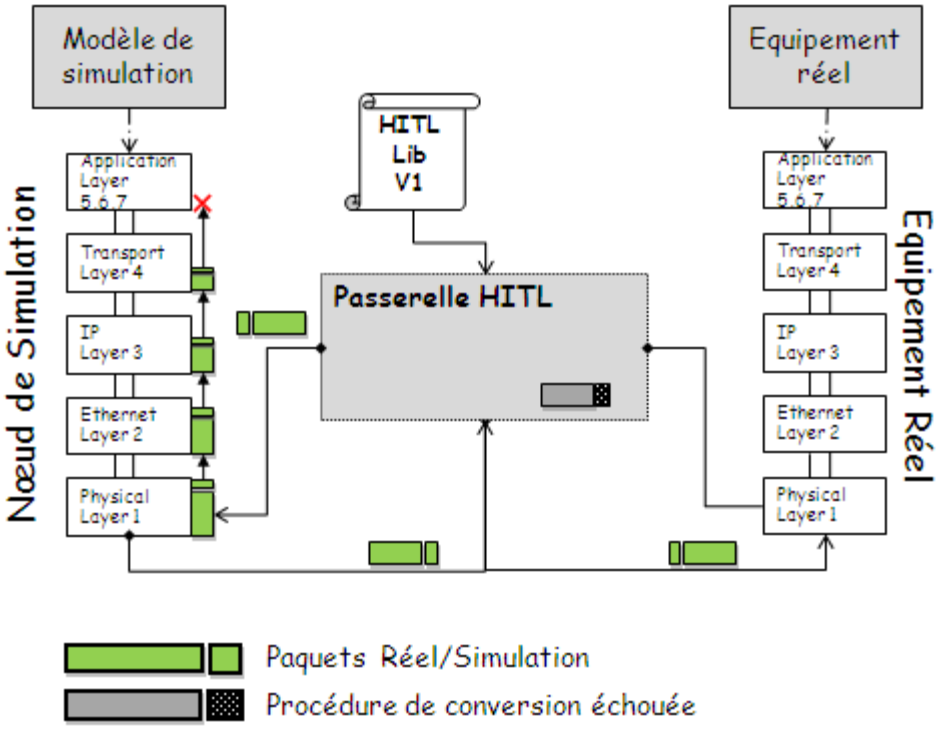


Figure 5.11 Test de l'implémentation de la première version de la librairie HITL

5.5.5.4 Deuxième version de la librairie HITL et résultat d'implémentation

Au niveau de la couche de transport, un débogage informatique nous a permis d'identifier la raison de l'arrêt de la décapsulation.

En effet, comme l'indique la figure 4.12, même les fonctions standards OpNet permettant la translation totale des paquets associés à la couche basse introduisent une carence dans la mise à jour des champs relatifs à la couche de transport. Par conséquent, cette dernière sera considérée comme partiellement convertie par les fonctions standards d'OpNet. Plus particulièrement, le champ **'data length'** appartenant à la zone de contrôle du segment TCP et renfermant la taille du paquet applicatif n'est pas mis à jour lors de l'appel aux fonctions de translation standard d'OpNet.

Par conséquent après réception d'un paquet provenant de la passerelle HITL, la couche TCP procède au contrôle du contenu du segment TCP reçu et détecte une défaillance du contenu du message de communication du fait que le champ **'data length'** n'est pas conforme à la taille de données associée à l'application. Ceci conduit à l'arrêt de la décapsulation au niveau TCP et le paquet ne sera pas envoyé vers la couche applicative.

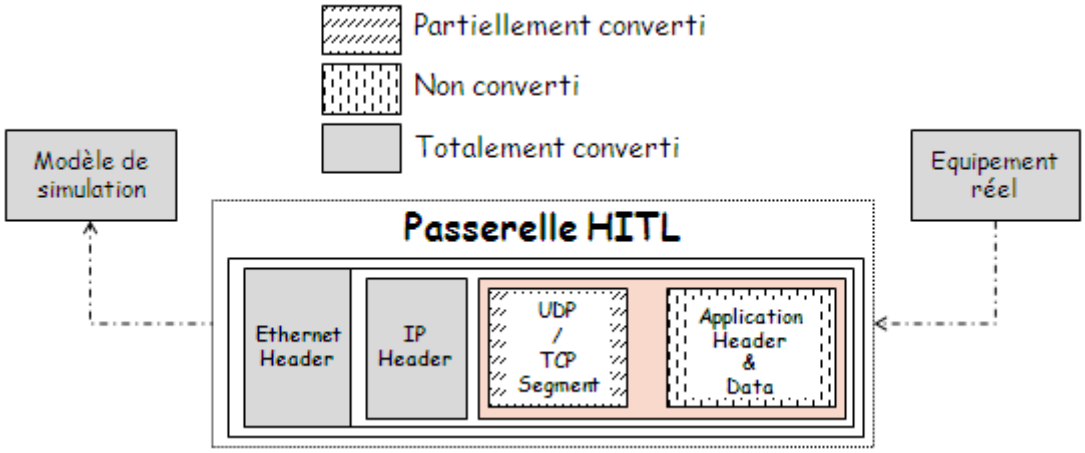


Figure 5.12 Test de l'implémentation de la première version de la librairie HITL

Pour y remédier, nous avons modifié la librairie HITL pour prendre en compte la mise à jour du champ TCP. Le détail informatique de cette modification est illustré dans l'annexe **{Annexe A}.** La figure 4.13 illustre le résultat de l'implémentation de la deuxième version de la librairie HITL qui a permis le déblocage de la transmission des paquets dans le sens réel-simulé.

Toutefois, comme le montre cette figure, cette version n'a toujours pas permis la transmission dans le sens simulé réel. Nous avons alors dû opérer une migration de la librairie HITL vers la version finale illustrée dans le paragraphe suivant.

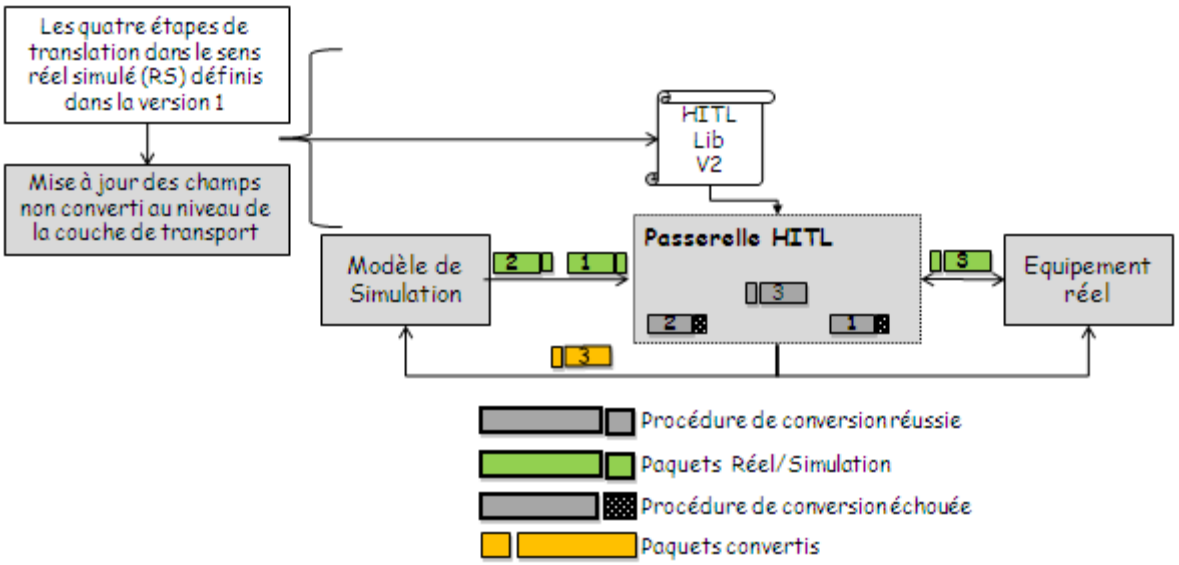


Figure 5.13 HITL Lib version 2

5.5.5.5 Version finale de la librairie HITL et résultat d'implémentation

Lorsqu'une application, associée à un modèle de simulation, procède à l'envoi d'un paquet, celui-ci subit une procédure de segmentation au niveau de la couche de transport. Il sera ainsi découpé en plusieurs sous-paquets, tout en respectant le paramètre TCP Maximum Segment Size 'MSS'.

La segmentation aura généralement lieu lorsque la taille du paquet applicatif dépassera la valeur MSS.

Informatiquement parlant, OpNet Modeler attribue un type informatique aux segments qui sont différents de celui d'un paquet non segmenté. Plus spécifiquement, un paquet segmenté est de type **'Packet*'** tandis qu'un paquet non-segmenté est de type **'segment*'.**

Les fonctions développées dans les versions 1 et 2 de la librairie HITL traitent le paquet de simulation reçu comme étant de type **'Packet*'**. En effet, lors de ces tests, nous avons considéré que l'application n'envoie pas un paquet d'une taille plus grande que la valeur MSS définie dans la couche TCP OpNet.

Toutefois, on a constaté qu'après avoir effectué des débogages via la console ODB, le paquet applicatif, quelle que soit sa taille, passe en type **'segment*'**. Ce passage est imposé par la couche TCP d'OpNet.

Ce qui fait que lors de l'appel des fonctions informatiques développées dans les versions 1 et 2 pour la translation, celles-ci ont signalé une erreur comme quoi elles ne parviennent pas à traiter le paquet. Cela a contribué à l'échec de la translation dans le sens simulé-réel.

Pour cela, nous avons ajouté une procédure dans la version 3 en vue d'effectuer la translation du paquet de simulation reçu, considéré de type **'segment*'**.Les détails de la programmation de cette procédure sont expliqués dans l'annexe {Annexe A}**.**

La figure 4.14 répertorie les résultats de l'implémentation de la version finale de la librairie HITL.

Comme indiqué sur cette figure, un bloc de fonction supplémentaire, qui transforme le type informatique, a été ajouté aux deux blocs de fonctions des versions 1 et 2 afin de créer cette version finale.

Cette dernière version a permis un fonctionnement total de la passerelle, qui peut désormais être utilisée pour échanger des communications entre les équipements réels et les modèles de simulation.

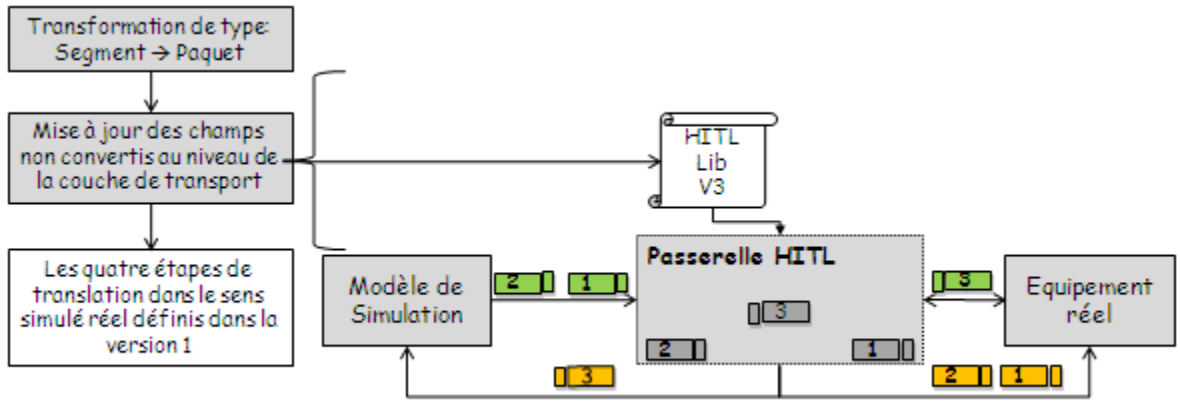


Figure 5.14 HITL Lib Version Finale

5.6 Module SITL

Le SITL est un module qui ne partage pas de composant avec le logiciel de simulation. Son but essentiel est la mise à disposition de fonctionnalités avancées pour réaliser des simulations.

Ces fonctionnalités ne peuvent pas être assurées si le logiciel de simulation est en mode 'Standalone'. La création de scénarios de simulation dynamique et l'accès aux variables des modèles en mode exécution sont deux de ces fonctionnalités.

La première partie de ce paragraphe fait état des fonctionnalités qui nous ont mené à la création de ce module.

La deuxième partie traite les difficultés imposées par le logiciel de simulation pour la mise en place d'une telle solution.

La troisième partie expose la solution technique choisie dans notre étude pour développer ce module.

Enfin, une application de test est indiquée afin de montrer la validation de fonctionnement du module.

5.6.1 Fonctionnalités requises pour une simulation avancée

Les deux fonctionnalités listées ci-dessous visent à faciliter les vérifications statique et dynamique expliquées dans {chapitre 3, paragraphe 3.5.2, paragraphe 3.5.3}. Ces fonctionnalités exigent la présence d'une interface utilisateur, qui a accès aux variables des modèles de simulation en mode exécution.

5.6.1.1 Interface utilisateur pour les phases de vérification et de validation

Après la phase de modélisation, une étape essentielle, qui permet de valider les modèles, doit être mise en place afin d'augmenter la confiance dans les modèles développés. Plus particulièrement, cette étape consiste en la validation de l'implémentation des services du protocole de communication dans les modèles de simulation.

Pour assurer ce point, une interface utilisateur permettant aux utilisateurs d'avoir un accès direct sur les variables de la simulation en mode exécution doit être mise en place.

Cette interface devra être capable d'effectuer des changements de valeurs de variables de simulation afin de générer les services évènementiels de communication à tester.

Pour la même raison, cette interface doit exister pour faciliter la vérification statique. En effet, lors de cette phase, il est amplement recommandé que les services de communication évènementiels soient générés dynamiquement sans pour autant devoir arrêter la simulation pour reconfigurer les modèles.

Pour assurer ce dialogue entre l'utilisateur et la simulation, l'interface utilisateur doit pouvoir totalement accéder à la base de données interne aux modèles de simulation.

La figure 4.15 illustre les deux intérêts déjà exposés pour cette interface utilisateur:

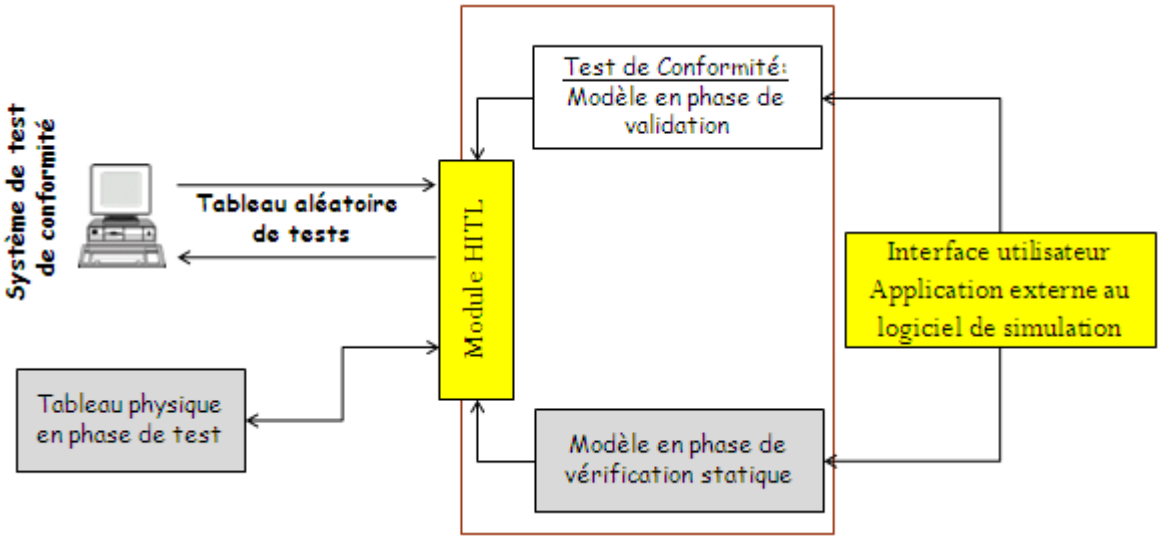


Figure 5.15 Utilité de l'Interface utilisateur dans la chaîne de vérification

5.6.1.2 Création des scénarios dynamiques

Comme expliqué dans {chapitre 3, paragraphe 3.5.3}, une vérification dynamique doit être établie en FAT pour réduire le temps de mise en service des architectures basées sur les nouveaux standards de communication. Cette vérification doit être basée sur une évaluation pointue et détaillée des performances du réseau IEC 61850.

Par conséquent, plusieurs études de performance, dont chacune est associée à une configuration de scénario d'échanges de trafics réseaux, doivent être réalisées sur les signaux de communication IEC 61850.

La création d'un scénario d'échange de trafic est généralement accomplie par la configuration d'attributs des modèles de simulation. Le logiciel de simulation permet de configurer ces modèles d'une manière statique. La désignation statique signifie que les modèles doivent être configurés avant le lancement de la simulation. Ainsi, une fois les valeurs des attributs définies, arrêter la simulation et opérer la modification est la seule méthode pour les modifier. Au lancement de la simulation les valeurs des attributs seront généralement reçues par les variables internes des modèles afin de générer les trafics de communication.

Toutefois, l'utilisation de cet aspect de configuration statique des scénarios engendre un impact négatif au niveau du temps requis pour la génération des performances liée à la vérification dynamique. Ceci est dû au nombre de modèles et au nombre de services existant dans chaque modèle.

Pour cela, une fonctionnalité avancée de simulation doit être mise en place dans le but de créer des scénarios de type dynamique. La conséquence de cette fonctionnalité est recoupée avec celle exposée précédemment par le fait que les variables internes aux modèles de simulation doivent être accessibles en mode exécution.

Ceci dit, l'interface utilisateur exposée auparavant pourra aussi être utilisée afin d'assurer la création de scénarios dynamiques pour les échanges de trafics de simulation.

La figure 4.16 expose la différence entre les deux modes de génération de scénarios de simulation :

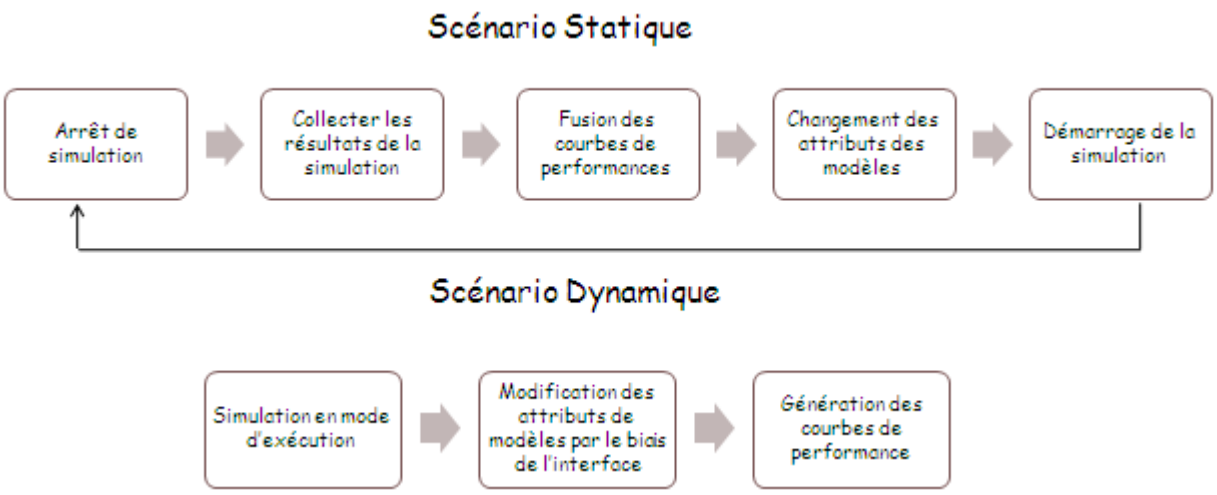


Figure 5.16 Etape de création d'un scénario dynamique et statique

Comme indiqué sur la figure 4.16, le laps de temps dû à l'arrêt de la simulation et à la fusion des courbes de performance sera réduit de 'n' fois. La valeur du paramètre 'n' correspond au nombre de changements effectués sur les attributs des modèles de simulation. Cette valeur est d'autant plus élevée que le nombre d'équipements présents dans l'installation, et leurs services de communication, sont élevés.

5.6.2 Problématique liée au logiciel de simulation 'OpNet'

Le développement de cette interface utilisateur nécessite la présence d'une application externe au logiciel de simulation, qui puisse accéder aux variables de simulation en mode exécution.

Néanmoins, le logiciel de simulation OpNet Modeler ne permet pas cet accès. En effet, 3 types de variables existent dans ce logiciel :

- Variable d'états
- Variable temporaires
- Variable globales

Ces variables sont simplement accessibles par des fonctions OpNet appartenant aux environnements suivants :

- Process Model
- Function Block (le bloc des fonctions associées au Process)
- External OpNet C File (fichier C OpNet)

Le tableau ci-dessous liste les fonctions qui peuvent accéder aux variables de simulation.

Tableau 5.2: Accessibilité des variables par les différents types de fonctions

		Visibilité des variables OpNet par les fonctions :			
		Process Model	**Function Block**	**External OpNet C File**	**Applications externes**
Type of variables	**Temporaire**	**X**			
	Etat	**X**	**X**		
	Global	**X**	**X**	**X**	

De fait, on peut conclure du tableau ci-dessus que seules les fonctions OpNet ont la possibilité d'accéder aux variables de l'environnement de la simulation.

Etant donné que l'interface utilisateur est considérée comme une application externe, elle ne pourra pas accéder aux variables de l'environnement de simulation.

Pour cela, le paragraphe suivant exposera le développement d'un module appelé 'SITL' qui a permis de contourner le problème.

5.6.3 Constitution du Module SITL

Etant donné qu'aucune application externe ne peut avoir un accès direct aux variables de la simulation en mode exécution, nous avons eu recours à une solution intitulée Software In The Loop **SITL**, qui permet de mettre en place un fichier tiers assurant l'interface entre les applications externes et le logiciel de simulation. Ce fichier est accédé en mode d'écriture par les applications externes et en mode de lecture par les modèles de simulation en exécution. A l'issue de cet accès, les modèles mettent à jour leurs variables en fonction des informations contenues dans ce fichier tiers (i.e. écrites par l'application externe).

Trois composants sont à l'origine du développement de ce module :

- Deux librairies précompilées en langage C
- Une base de données type SQL
- Une application externe jouant le rôle d'interface entre la simulation et l'utilisateur

La base de données SQL est utilisée en tant qu'élément intermédiaire entre le logiciel de simulation et l'application externe.

Ainsi, cette base de données sera un fichier accessible en mode lecture par l'environnement de la simulation et en mode d'écriture par l'application externe. Les commandes envoyées par l'application externe seront formulées par une modification des champs dans la base de données, qui sera par la suite traduite par une mise à jour des variables de simulation associées à ces champs.

Pour permettre un accès en mode lecture, le logiciel de simulation doit intégrer deux librairies de fonctions compilées. La première librairie est une librairie système et la deuxième est développée par l'utilisateur.

Le but de la librairie système revient à assurer un dialogue de communication entre le logiciel de simulation se comportant en tant que client et le serveur de la base de données SQL.

La librairie système assure un dialogue de communication entre le logiciel de simulation qui se comporte comme un client et le serveur de la base de données SQL. Le protocole de communication utilisé pour l'échange des requêtes SQL est ODBC. La librairie implémentée a pour titre odbc32.lib.

La librairie SITL est une librairie développée dans notre étude qui fournit des fonctions de lecture et/ou écriture dans une base de données SQL.

De même, ces deux librairies sont implémentées dans l'application externe afin d'accéder à l'écriture des champs dans la base de données SQL.

La figure 4.17 montre l'interconnexion entre les différents constituants du module SITL.

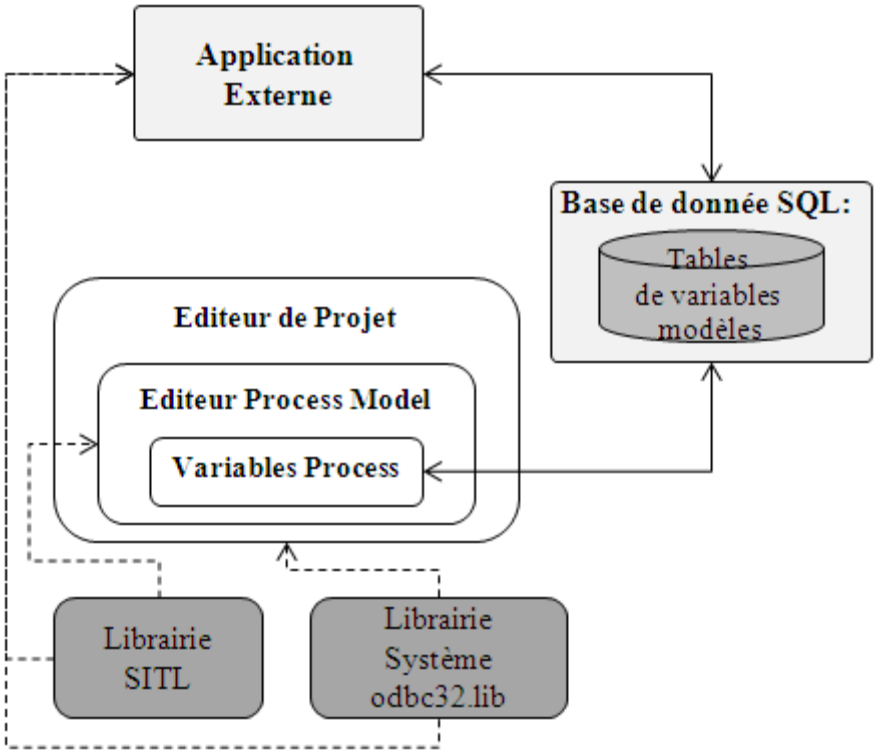


Figure 5.17 Constituant du module SITL

5.6.4 Test de SITL

Une simple application a été développée dans notre étude pour tester le Module SITL. Elle contient trois valeurs analogiques (courant, tension, fréquence) et une valeur tout ou rien (statut de l'élément de coupure).

Un simple modèle de Process indiqué sur la figure 4.18 est établi pour récupérer les valeurs électriques envoyées par l'application externe. La base de données est formée d'une simple table contenant 4 colonnes correspondant aux valeurs envoyées par l'application externe et une colonne indiquant l'apparition d'un changement.

Dans l'étape **'Init'**, les valeurs électriques sont initialisées et la connexion à la base de données est établie.

La transition vers l'étape **'Wait'** se fait en mode non conditionné (Trait non pointillé dans la Fig. 4.18). Dans cette étape, un appel aux fonctions de la librairie SITL sera effectué afin de lire périodiquement les champs de la base de données.

Lorsque l'application externe envoie une nouvelle valeur, le changement sera détecté au niveau de l'étape 'Wait', qui déclenchera un évènement permettant la transition vers l'une des quatre étapes correspondant à ce changement.

Dans l'exemple, nous avons exposé un changement de la valeur de la tension décrite sur la console OpNet montrée en bas de la figure 5.18.

La mise à jour des variables de simulation est effectuée dans les étapes **voltage**, **current**, **frequency** et **status**. Les valeurs sont affichées sur la console OpNet Debugger lors de chaque mise à jour.

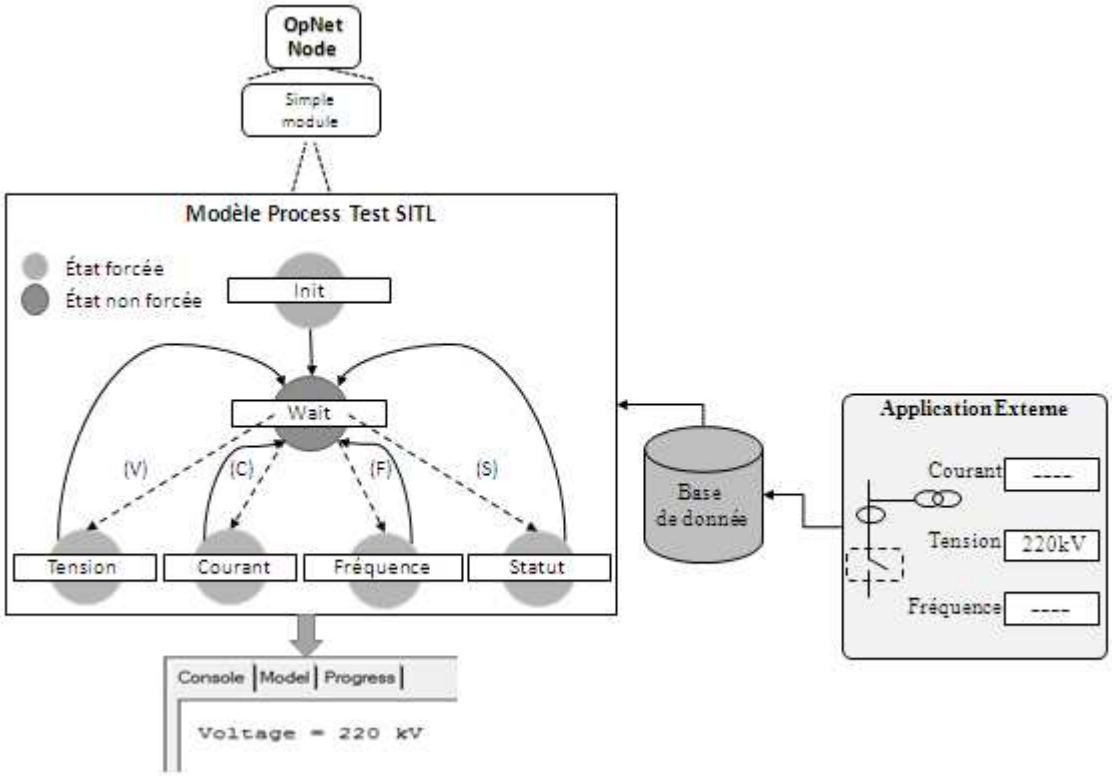


Figure 5.18 Application de test du module SITL

5.7 Conclusion

Les modules expliqués au cours de ce chapitre ont contribué au développement d'une plateforme de Co Simulation opérationnelle.

Cette plateforme peut désormais être utilisée pour réaliser des simulations avancées répondant aux exigences de test et de vérification décrites dans {chapitre 3, paragraphe 3.5}. Elle sera ainsi considérée comme un moyen unique pour les différentes phases de test et de vérification.

Les chapitres suivants vont montrer que le module SITL sera essentiel pour permettre le contrôle des modèles sans arrêter la simulation. Cet aspect autorise une modification dynamique des flux de données générés par les modèles de simulation ce qui facilite la phase de vérification d'un SDC complexe type IEC 61850.

A son tour, le module HITL sera considéré comme le module essentiel accompagnant toutes les phases de vérifications. En effet, le but de ce module est de permettre le remplacement des équipements réels inexistant en phase de vérification par des modèles d'équipements pouvant se connecter à la partie d'équipements réel en phase de vérification. Ce module jouera également un rôle crucial au niveau des phases de validation des modèles de simulation en connectant ces derniers à des systèmes réels permettant de réaliser les tests de conformité.

Chapitre 6

Modélisation protocolaires et validation des modèles de simulation

6.1 Introduction

Le chapitre précédent a montré les développements effectués pour rendre opérationnelle notre plateforme de Co-Simulation. De plus, nous avons illustré dans {chapitre 4, paragraphe 4.5} les intérêts globaux que peut apporter cette plateforme dans le cadre de notre travail de recherche.

Le présent chapitre va présenter les premiers travaux effectués dans le but de concrétiser les possibilités de cette plateforme de Co-Simulation. En particulier, nous allons montrer la méthodologie permettant la validation des modèles de communication en utilisant les modules de la plateforme. L'aspect principal concerne la validation de la conformité des fonctionnalités de communication, implémentées dans les couches applicatives des modèles, par rapport à la spécification du protocole de communication choisi.

Les trois premiers paragraphes illustrent le développement d'un modèle de communication basé sur un protocole applicatif industriel très répandu et très connu puisqu'il s'agit du ModBus/IP ; ceci lui permettra de nous servir d'exemple efficace pour valider le premier intérêt de la plateforme. Nous allons montrer tout d'abord le développement d'un bloc process OpNet permettant la gestion des connexions TCP/IP. Ce bloc a été développé en utilisant des attributs de modèles du logiciel OpNet Modeler pour servir à simplifier la complexité de programmation des modèles.

Le dernier paragraphe montre l'architecture de Co-Simulation et la stratégie permettant la validation de l'implémentation des fonctionnalités du protocole ModBus/IP dans notre modèle de communication. Les résultats de validation sont montrés à la fin de ce chapitre.

6.2 Gestion des connexions TCP/IP dans les modèles de simulation

Pour assurer une transition des messages entre les couches de communication d'un modèle de simulation (i.e. module), des échanges de commandes et des indications de traitement de messages doivent être établis entre les couches.

OpNet Modeler implémente les protocoles de communication standard du modèle OSI dans les couches de communication des nœuds de simulation en partant de la couche physique jusqu'à la couche transport. Un exemple de ces protocoles est le standard TCP/IP. L'échange des commandes et des indications entre les couches basses (i.e. par rapport à la couche application) est assuré automatiquement par le logiciel OpNet lors de l'implémentation de ces couches dans les nœuds de simulation.

Toutefois, l'implémentation des protocoles de communication dans la couche applicative ainsi que l'échange des commandes et des indications entre cette couche et les couches basses du modèle OSI n'est pas assurée par ce logiciel de simulation. Ceci est dû au fait de la diversité des protocoles existant au niveau de la couche application.

Par conséquent, la programmation des couches applicatives ainsi que leurs cohabitations avec les couches basses, telle que la couche de transport implémentant le protocole standard TCP ou UDP, doivent être développés en fonction des besoins. L'échange des commandes et d'indication entre les couches de communication est crucial à mettre en œuvre pour n'importe

quelle modélisation protocolaire du fait qu'elles permettent d'une part la gestion des connexions des nœuds de simulation et d'autre part l'échanges des messages de communication.

Le premier sous paragraphe montre le principe ainsi que les différentes fonctions permettant l'échange de commandes et d'indications entre une couche applicative et une couche TCP d'un nœud de simulation. Le deuxième sous-paragraphe explique la construction d'un bloc process qui a été développé dans le but d'assurer d'une part la gestion des connexions et de simplifier d'autre part le développement des modèles de simulation.

6.2.1 Principe de cohabitation entre la couche applicative et la couche transport OpNet

Plusieurs **API 'Application Protocol Interface'** sont livrées par le logiciel de simulation OpNet pour assurer la tâche de cohabitation. Ces API mettent à la disposition des couches applicatives des fonctions informatiques permettant d'envoyer des commandes et de recevoir des indications vers les couches basses du nœud de simulation.

Etant donné que le protocole ModBus/IP fonctionne au-delà de la couche transport TCP, les deux librairies de fonction **'tcp_api_v3'** et **'ip_addr_v4'** (figure 5.1), livrées par le logiciel de simulation OpNet, ont été utilisés dans notre étude pour assurer la cohabitation entre la couche applicative ModBus et la couche transport TCP.

Comme le montre la figure 5.1, pour assurer une gestion des connexions TCP/IP, plusieurs étapes doivent être mise en place en respectant une chronologie de cohabitation entre la couche applicative et la couche transport.

La première étape consiste à faire appel à une fonction intitulée 'Register' qui permet de générer une structure contenant les caractéristiques des deux couches de communication. Cette structure est utilisée en tant que paramètre pour toutes les autres fonctions de commandes et de signalisations d'informations entre ces deux couches.

Le mécanisme de connexion TCP intitulé **'Three Way Hand Shaking'** consiste à ce qu'un client envoie une requête de synchronisation vers un serveur qui répondra par la suite par un acquittement de cette requête pour qu'à la fin le client envoie l'acquittement final indiquant que la connexion a bien été établie. Ce mécanisme est géré par la couche de transport du serveur ou du client suite à des commandes de connexions envoyées par la couche applicative de ces derniers (figure 5.1).

Au niveau serveur, la commande de connexion envoyée par la couche applicative vers la couche TCP doit aboutir à une ouverture d'une connexion passive sur le port de communication correspondant au protocole implémenté dans l'application. A l'issue de cette demande, la couche de transport renvoie une information à l'application indiquant si le serveur a accepté la commande envoyée.

Au niveau client, l'application doit envoyer une commande vers la couche TCP dans le but de se synchroniser avec le serveur considéré à l'écoute. A l'issue de cette commande, la couche transport du client envoie la demande de synchronisation au serveur et informe par la suite la couche applicative du résultat de cette synchronisation.

La fonction **'tcp_connection_open_with_options ()'** appartenant à la librairie **'tcp_api_v3'** permet d'assurer ces deux types de commandes. Le type de connexion (i.e. active ou passive)

constitue l'un des paramètres essentiels de cette fonction. Les autres paramètres permettent d'indiquer les adresses sources et destination des deux équipements client et serveur.

Les adresses IP dans OpNet Modeler sont attribués à un type informatique intitulé **'InetT_Address'**. Une fonction appartenant à la librairie **'ip_addr_v4'** permet la génération d'une adresse de ce type en partant d'une adresse exprimée en une chaîne de caractères.

Une fois la connexion établie, l'échange des paquets de communication protocolaire entre le client et le serveur peut avoir lieu. L'envoi d'un paquet de communication depuis un modèle de simulation est accompli par la fonction **'tcp_data_send'** de la librairie **'tcp_api_v3'**. Quant à la réception des paquets, la couche applicative du modèle doit en premier temps informer la couche de transport, par l'intermédiaire de la fonction **'tcp_receive_command_send'** de la librairie **'tcp_api_v3',** de sa capacité à recevoir. A l'issue de cette commande la couche de transport envoie automatiquement le paquet reçu vers la couche applicative.

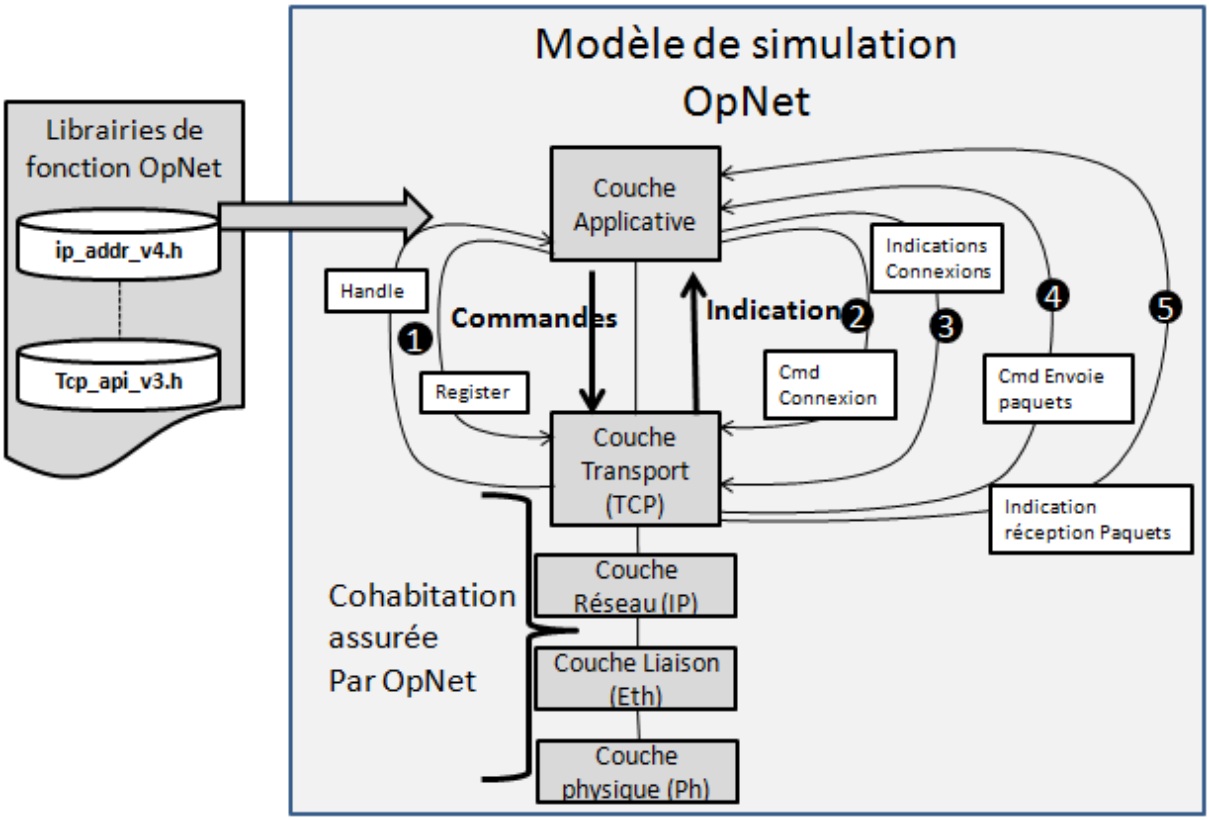


Figure 6.1 Commandes et indications échangées entre la couche applicative et la couche transport pour la gestion des connexions TCP/IP

6.2.2 Attributs de modèles et généralisation de la gestion de connexions TCP/IP

La cohabitation entre la couche application et la couche de transport est un point essentiel à intégrer dans le développement de n'importe quel modèle de simulation utilisant le profil TCP/IP. Il a été montré au cours du paragraphe précédent que cet interfaçage nécessite l'utilisation et le paramétrage de plusieurs fonctions OpNet appartenant à deux librairies de l'API TCP.

Une généralisation de cette cohabitation a été effectuée dans notre étude en créant un bloc process intitulé **'Connexion Establishment'**. Le but de ce bloc est de simplifier la tâche de gestion de connexions TCP/IP des modèles de simulation pour les rendre configurables par l'intermédiaire des attributs de connexion créés dans la couche applicative (figure 5.2).

En effet, chaque couche de communication appartenant à un modèle de simulation est associée à plusieurs attributs permettant la configuration de son comportement. Ces attributs peuvent être de type simple ou composé. Un exemple des attributs simple d'une couche est le type du **'Process'** associé à cette couche. Les développeurs Modeler peuvent rajouter des **'attributs utilisateurs'** pour donner plus de flexibilité à la configuration de leur modèles. Dans notre étude, on a utilisé cette notion pour créer deux attributs composés intitulés **'Active Connection'** et **'Passive Connection'** (figure 5.2). Ces attributs sont utilisés par le Process **'Connexion Establishment'** dans le but de rendre la procédure de gestion de connexion TCP complètement configurable (figure 5.2).

Le premier attribut permet de définir le nombre ainsi que toutes les caractéristiques d'une connexion active telles que l'adresse IP source/destination, le numéro de port sources/destination (figure 5.2). Tandis que le deuxième permet de définir le nombre et les caractéristiques d'une connexion passive TCP. Un modèle peut jouer le rôle d'un client et d'un serveur en même temps.

La figure 5.2 montre les différentes étapes constituant le bloc **'Connexion Establishment'**.

L'étape **'Initialization'** permet d'extraire les informations contenues dans les attributs de connexion du nœud de simulation. Les conditions **'a'**, **'b'** et **'c'**, montrées sur la figure 5.2, sont implémentées dans les transitions de process de manière à ce que ce dernier puisse établir toutes les connexions passives et actives configurées au niveau de ces attributs. La répartition de ces transitions montrent que le process établi en premier les connexions passives en passant par l'étape **'Passive_con'** pour ensuite transiter vers l'étape **'Active_Con'** où il procède à l'envoi des commandes de synchronisation pour ouvrir les connexions actives. Chacune de ces deux dernières étapes contient la fonction **'tcp_connection_open_with_options ()'** expliquée auparavant. Les deux étapes sont de type non forcées du fait qu'elles doivent bloquer le Process en attendant le résultat de leur commande. Le déblocage du 'Process' de la couche application aura lieu lors de la réception d'une interruption envoyée par la couche transport. Cette interruption indiquera le résultat de la commande qui sera affiché dans l'étape **'Connection Status'**.

La transition du bloc de l'établissement de connexion **'Connexion Establishment'** vers le reste du Process de la couche applicative est conditionnée par la variable **(!c)** indiquant que toutes les connexions définies dans les attributs du modèle ont été établies avec succès.

Le bloc **'Connexion Establishment'** ainsi que les attributs de connexion sont développés dans notre étude dans le but de simplifier le développement de la modélisation au niveau de la couche applicative. Par conséquent, nous pouvons remarquer que ce bloc sera utilisé dans tous les modèles de simulation développés dans notre étude.

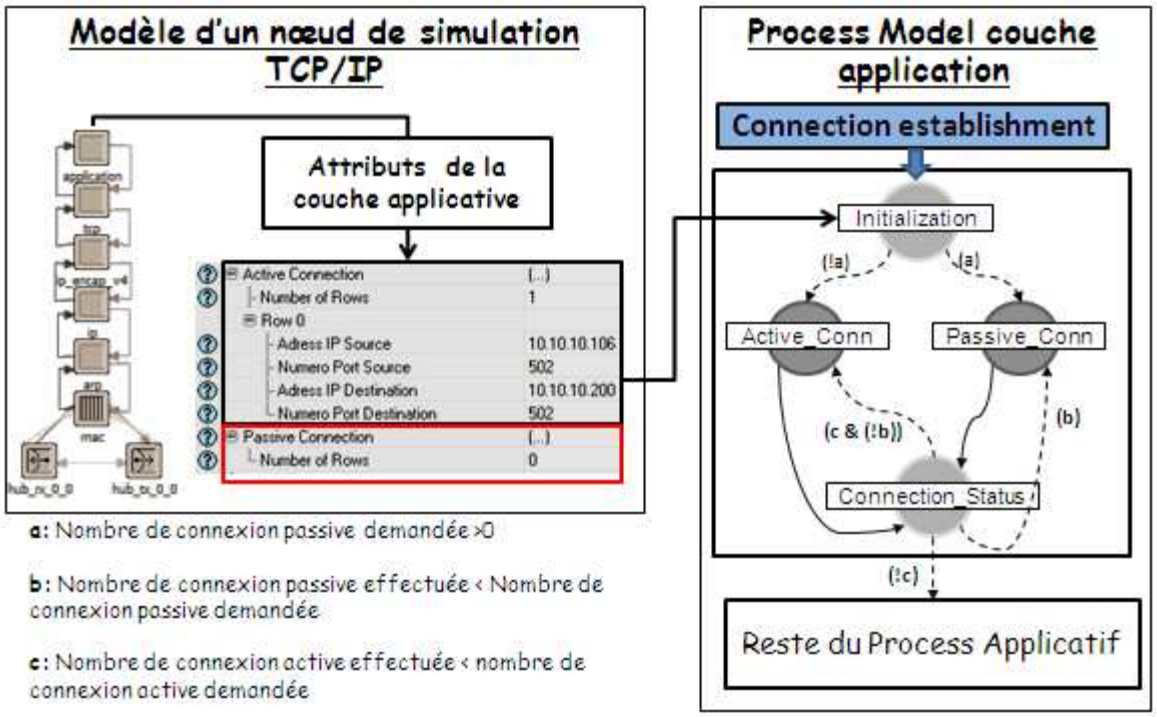


Figure 6.2 Composition du process connexion establishment et interconnexions avec les attributs de la couche applicative

6.3 Protocole de communication ModBus

ModBus est un protocole de communication qui a été proposé par Modicon (actuellement Schneider Electric) en 1979. Ce protocole était conçu, dans un premier temps, pour fonctionner sur une couche physique OSI de type série basée sur des connecteurs et des câbles RS232 ou RS485.

Les protocoles de communication utilisant cette couche physique sont généralement basés sur le principe maître/esclave pour la génération des flux de communication. Le maître est le seul élément actif à l'origine de la génération de ces flux. Les esclaves attendent une requête de communication protocolaire venant du maître avant d'envoyer leurs réponses.

Une variante ModBus, intitulée 'ModBus/IP', a été développée ces dernières années dans le but de permettre l'utilisation de ce protocole sur un réseau de communication Ethernet. Cette variante constitue l'un des protocoles de communication les plus répandus dans les architectures de communication d'un SDEE [Goraj et al, 2006]. La différence entre les composants constituant une architecture ModBus série et une architecture ModBus/IP pour un SDEE a été décrite dans [Haffar et al, 2007].

Nous avons choisi ce protocole de communication pour la réalisation de nos premiers travaux de modélisation et de Co-Simulation du fait de son aspect très répandu dans l'industrie. Ce point nous facilite la tâche de réalisation et de tests des architectures Co-Simulées qui vont nous permettre de montrer et valider les intérêts qu'apportent notre plateforme dans le domaine de la validation de modèles et d'évaluation de performance des architectures de communication mixtes (composées d'équipements virtuels et réels).

Par ailleurs, ce protocole est 'Open Source' ou 'libre de droit', ce qui permet d'avoir une connaissance complète du format des messages constituant ses fonctionnalités de communication. Ceci permet de faciliter la tâche de l'implémentation de ces fonctionnalités de communication dans les couches applicatives des modèles de simulation d'OpNet.

6.4 Construction du modèle du serveur ModBus/IP

Le premier modèle de communication développé dans notre étude est le serveur ModBus/IP. Ce serveur est implémenté dans divers produits industriels tels que les automates programmables et les unités de protections.

Le modèle du **'Process'** montré dans la figure 5.3 illustre le contenu de la couche applicative de ce modèle [Haffar et al, 2010d].

ModBus prévoit quatre fonctionnalités de communication de base permettant l'accès aux données appartenant à la mémoire du composant implémentant ce protocole {Annexe B}. Ces fonctionnalités sont : Lecture des informations analogiques, Lecture des informations binaires, Ecriture des informations analogiques et Ecriture des informations binaires. La mémoire d'un serveur ModBus est généralement divisée en deux catégories : la première contient les informations binaires, et la deuxième contient les informations numériques.

Le Process de la couche applicative du modèle du serveur est divisé en trois niveaux (figure 5.3). Le premier niveau implémente une étape d'initialisation et le bloc process permettant la gestion des connexions TCP/IP expliqué dans {chapitre 5, paragraphe 5.2.2}. L'étape **'Initialisaton'** permet de donner des valeurs initiales des données contenues dans les deux mémoires (binaires et analogique du modèle serveur). Le bloc **'Connexion Establishemnt'** vise à initier une connexion passive en mettant le port de communication numéro 502 (dédié au protocole ModBus) à l'écoute des synchronisations venant des clients ModBus.

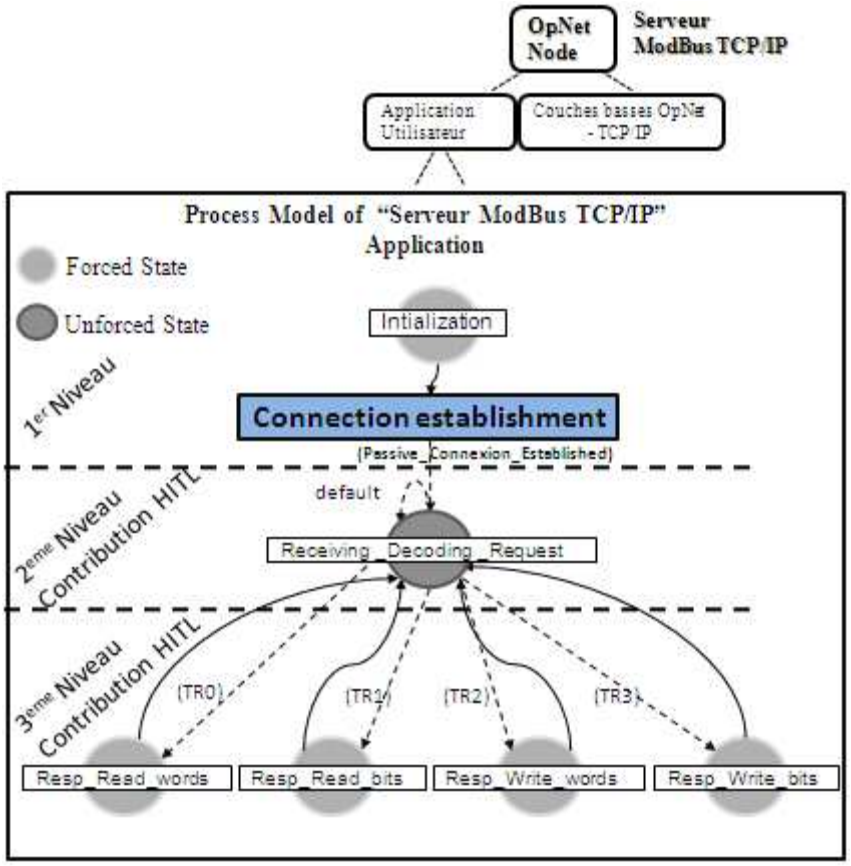


Figure 6.3 Process Model Serveur ModBus TCP/IP

Le deuxième niveau est composé d'une seule étape non forcée ayant pour rôle de bloquer le Process en attendant une interruption envoyée par la couche TCP du modèle indiquant l'arrivée d'un nouveau message de communication. Cette interruption est de type 'OPC_INTRPT_STRM', elle est implémentée dans les quatre transitions TR0, TR1, TR2 et TR3, issues de l'étape du niveau 2 (figure 5.3). A la détection d'une telle interruption, le 'Process' applicatif fait appel à des fonctions OpNet afin d'identifier le paquet reçu {Annexe B}. Si le résultat de ces fonctions indique que le paquet est de type ModBus, le Process transite vers le niveau 3. Dans le cas contraire, la transition **'default'** sera active et le Process reste bloqué au niveau 2.

La différence entre les quatre transitions, issues du niveau 2 et allant vers le niveau 3, persiste au niveau du type de la fonctionnalité de communication ModBus reçue. La transition **'TR0'** indique que la fonctionnalité de communication reçue est de type lecture informations analogiques. Les conditions **'TR1'**, **'TR2'** et **'TR3'** correspondent respectivement aux fonctionnalités de communication lecture informations binaires, écriture informations analogiques et écriture informations binaires.

Chacune de ces transitions dirige le Process applicatif vers une étape permettant la construction de la réponse correspondante à la requête reçue. Ces étapes utilisent des fonctions spécifiques OpNet illustrées dans l'annexe {Annexe B}. La fonction **'tcp_data_send'** est utilisée à la fin de chacune des étapes du niveau 3 afin de transmettre la réponse construite vers sa destination.

Ce qui est important à souligner, c'est que l'interaction entre le Process applicatif et le module HITL de la plateforme intervient aux niveaux 2 et 3 dans le but d'assurer un échange de communication entre le modèle construit et les équipements industriels appartenant au monde réel.

6.5 Architecture Co-Simulée pour la validation de la modélisation du protocole ModBus/IP serveur

Nous avons vu {chapitre 4, paragraphe 4.5.1} que plusieurs utilisations peuvent être envisagées pour la plateforme de Co-Simulation. Ce paragraphe va montrer le développement d'une architecture mixte, composée d'équipements réels et de modèles virtuels, et d'un scénario de communication ayant pour but de valider les fonctionnalités de communication ModBus implémentées dans le modèle du serveur. Ce modèle sera vu comme un Model Under Test 'MUT'.

La figure 5.4 montre les différents constituants et les différents types de liens entre les constituants de cette architecture. Cette dernière est composée du modèle serveur à tester 'MUT', du module HITL développé et des équipements appartenant à la zone réelle. L'interface de communication virtuelle du MUT est connectée à la passerelle HITL par l'intermédiaire d'un lien de communication pour autoriser l'échange de données entre le MUT et les équipements de la zone réelle. Cet échange sera assuré par l'intermédiaire de l'interface de communication de la machine de Co-Simulation qui est connectée par l'intermédiaire d'un lien logique à la passerelle HITL de la plateforme (figure 5.4). Finalement cette figure montre que pour le bon fonctionnement du module HITL, la librairie HITL Lib doit être implémentée dans la passerelle HITL (lien implémentation).

L'équipement existant dans la zone réelle est choisi de manière à ce qu'il puisse envoyer des tableaux de tests vers le MUT afin de valider les fonctionnalités de communication implémentées dans ce dernier. Par conséquent, le composant choisi doit être complémentaire au sens de la communication au MUT (serveur si le MUT est un client et réciproquement). Etant donné que notre MUT est un serveur, n'importe quel client du marché conforme au protocole de communication ModBus peut jouer le rôle de l'équipement de test. Ainsi, nous avons choisi dans notre étude le logiciel de supervision 'PCVUE' délivrée par la société 'ARCINFO'. PCVUE est un superviseur reconnu dans le monde industriel.

Une application de supervision a été développée de manière à ce qu'elle implémente les différentes fonctionnalités de communication à tester dans le MUT. La validation de ces fonctionnalités est traduite par le fait que le modèle puisse répondre à toutes les requêtes de communication envoyées par l'application de supervision.

Cette expérimentation a fait l'objet d'une démonstration publiée à la journée démonstrateur 2010 réalisée à Angers [Haffar et al, 2010d].

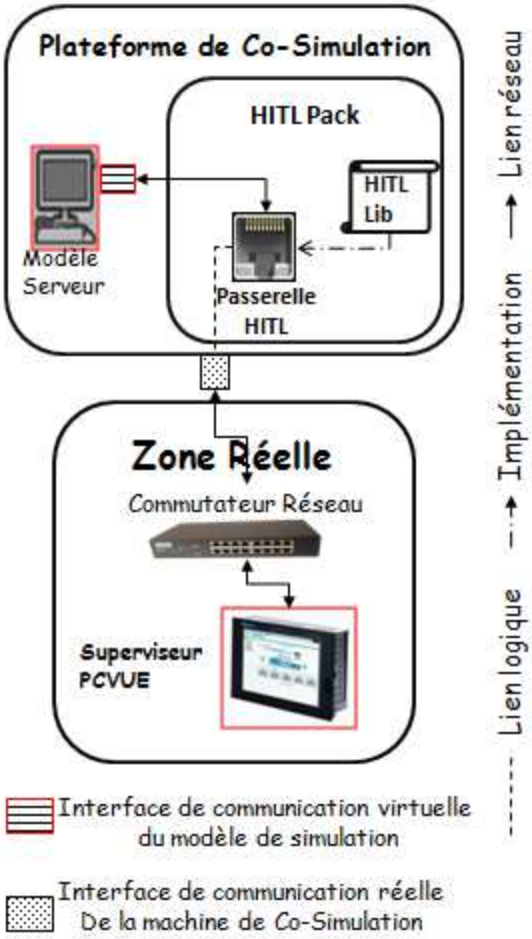


Figure 6.4 Constituants essentiel d'une architecture de validation du modèle serveur ModBus/IP

6.5.1 Scénario de communication et résultats de validation

La validation des fonctionnalités de communication implémentées dans le MUT serveur a pour but de vérifier le bon déroulement des échanges de communication d'écriture/lecture des informations binaires et numériques entre l'application de supervision conforme ModBus et le MUT.

Deux informations binaires situés aux adresses 4 et 5 et deux autres informations analogiques situées aux adresse 5 et 6 ont été choisies pour la validation des fonctionnalités de communication implémentées dans le MUT (figure 5.5).

Au démarrage de l'expérimentation, les informations du modèle serveur ont été initialisées comme suit : l'information localisée à l'adresse 4 est initialisée à '1', celle existant dans l'adresse 5 est initialisée à '0'. L'information numérique située à l'emplacement 5 de la table des informations analogiques du serveur a été initialisée à la valeur '10', celle existante à l'adresse 6 est initialisée à la valeur '20'.

L'application de supervision a été configurée de manière à envoyer deux trames de lecture périodiques des quatre informations choisies dans le serveur. La première trame est de type lecture n bits tandis que la deuxième est de type lecture n mots. Deux courbes de tendance montrées sur la figure 5.5 ont été implémentées dans cette application dans le but d'afficher l'évolution des informations du serveur obtenues à la suite de l'envoi des trames de lecture périodique.

La première courbe consiste à montrer les résultats des lectures périodiques des informations binaires 4 et 5 tandis que la deuxième affiche les résultats de lecture des informations numériques. Trois commandes ont été ajoutées dans l'application de supervision (figure 5.5) afin de lancer des demandes d'écriture des informations contenues dans le modèle serveur.

La commande 1 est une commande d'écriture des informations binaires. Cette commande est configurée de manière à inverser les valeurs des informations binaires 4 et 5 du serveur. Elle est envoyée à l'instant 'T1' après une minute et demie du démarrage de la simulation.

La commande 2 est une commande d'écriture des informations numériques. Elle est configurée de manière à écrire la valeur '30' dans le mot situé à l'adresse 5 et la valeur '10' dans le mot situé à l'adresse 6. Cette commande est envoyée à l'instant T2 après deux minutes et demie du démarrage de la simulation.

La commande 3 est une commande permettant d'envoyer deux requêtes d'écriture, une pour les informations numériques et une autre pour les informations binaires. Le but de cette commande est de remettre aux valeurs des informations serveurs à leurs états initiaux. Cette commande est envoyée à l'instant T3 après 3 minutes et demi du démarrage de la simulation.

Les deux courbes de tendance affichées dans le logiciel de supervision sont bien conformes au scénario de communication défini. Ceci nous permet de valider que le serveur a bien pu répondre aux requêtes de lecture envoyées par le client (i.e. les fonctionnalités de lecture ModBus sont bien implémentées dans le modèle serveur).

La validation des fonctionnalités d'écriture peut de même être déduite de ces deux courbes de supervision (figure 5.5). En effet, le changement des allures de ces deux courbes après les instants T1, T2 et T3 permet de valider que le MUT a bien pris en compte le changement des valeurs de ces variables internes après la réception des commandes d'écriture ModBus envoyées par le système de supervision.

Une autre façon de vérifier la validation de ces deux fonctionnalités d'écriture est illustrée sur la figure 5.6. Les deux courbes montrées sur cette figure correspondent aux enregistrements des statistiques des données contenues dans le MUT.

On peut remarquer de même dans cette figure qu'à l'issue des commandes d'écriture envoyées par le système de supervision aux instants T1, T2 et T3, le MUT a bien pris en compte ces requêtes en modifiant sa base de données interne en fonction des commandes d'écriture envoyées par le système de supervision. Ceci permet donc de confirmer que les fonctionnalités de communication ModBus permettant l'écriture des informations binaires et numérique ont été implémentées avec succès dans le MUT.

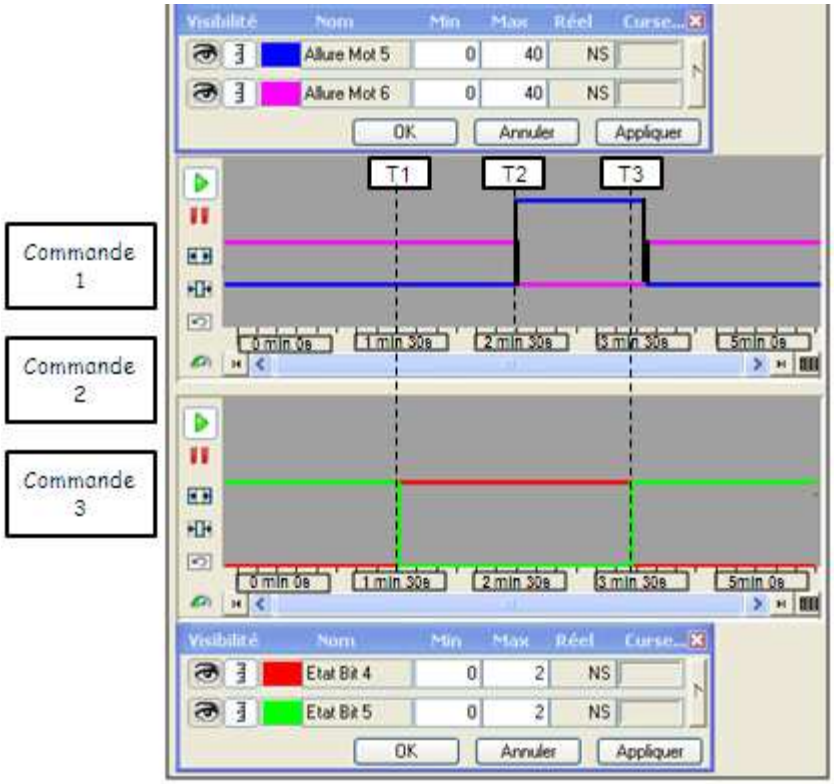


Figure 6.5 Allures des données du MUT affichées au niveau des courbes PCVUE - Validation de toutes les fonctionnalités de communication

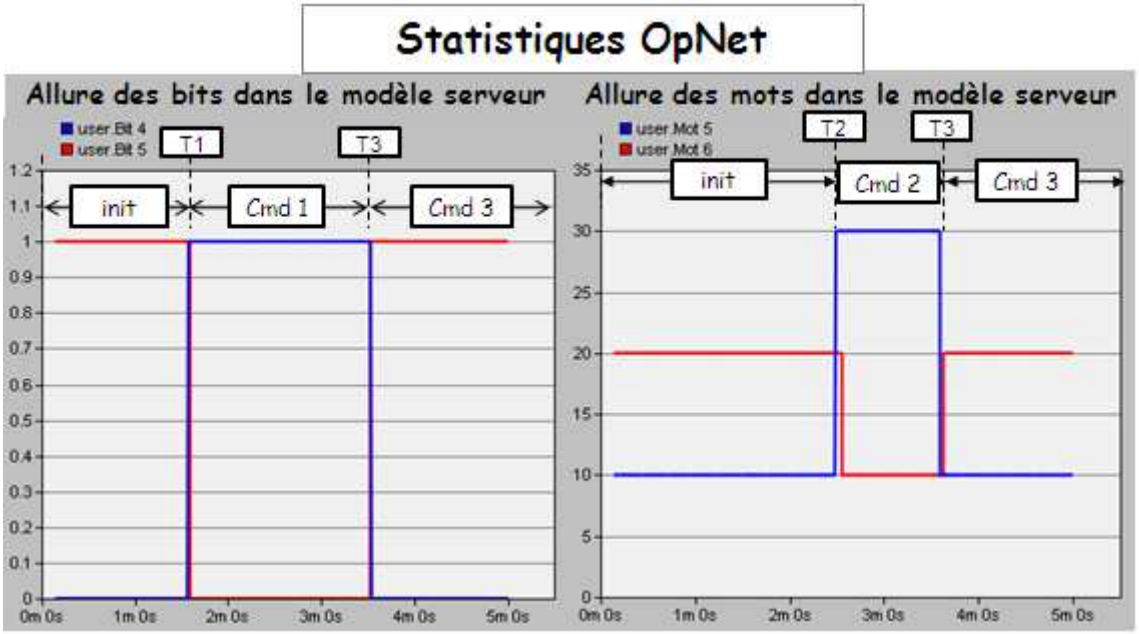


Figure 6.6 Allure des données MUT affichées au niveau des statistiques OpNet - Validation des fonctionnalités de communication d'écriture

6.6 Conclusion

Il a été montré dans {chapitre 3, paragraphe 3.5} que le but principal de la plateforme de Co-Simulation est de permettre la réalisation de la phase de vérification d'un SDC IEC 61850. Cette phase exige une validation des modèles de communication avant d'être implémentés dans une architecture mixte pour cohabiter avec les DUT en phase de FAT. La validation des modèles vient dans le sens de vérifier la conformité de l'implémentation des fonctionnalités de communication dans les couches applicatives des modèles, par rapport à la spécification du standard de communication. Cette validation est un point fortement recommandé par les industriels du fait qu'elle permet de hausser la confiance dans les modèles de communication construit dans un environnement virtuel.

Ce chapitre a montré, par le biais de la construction d'un modèle serveur implémentant un simple protocole de communication 'ModBus/IP', que notre plateforme de Co-Simulation peut servir à valider les modèles développés dans l'environnement de la simulation. Plus spécifiquement, cette validation passe par le module HITL de la plateforme de Co-Simulation qui permettra dans ce cas la connexion entre les systèmes de tests de conformité et les 'MUT' construits dans l'environnement de la simulation.

La validation du MUT ModBus/IP a été présentée à la fin de ce chapitre en connectant un système de supervision conforme au protocole ModBus/IP à notre modèle par le biais du module HITL. Ce système de supervision a joué le rôle du système de tests de conformité de ce simple protocole de communication en échangeant une séquence de communication pour permettre de valider les fonctionnalités de communication implémentées dans le MUT.

Les résultats de ce chapitre doivent être étendus pour la validation de l'implémentation du standard IEC 61850 dans les modèles de communication OpNet.

Chapitre 7

Evaluation de performance des architectures Co-Simulées

7.1 Introduction

Ce chapitre va présenter plusieurs Co-Simulations effectuées sur des architectures de communication mixte à base du protocole ModBus/IP. Le but de ces Co-Simulations revient d'une part à faire des études comparatives entre l'impact des modèles de communication et l'impact des équipements réels sur les performances d'une architecture de communication, et d'autre part à évaluer les retards qu'imposent les modules de la plateforme sur la performance d'une architecture Co-Simulée.

Les architectures Co-Simulées présentées dans ce chapitre sont symbolisées par des lettres majuscules et des indices associés à chaque lettre. La lettre majuscule représente le type de l'élément (**S** pour un élément **S**imulé et **R** pour un élément **R**éel), tandis que l'indice représente la nature de l'élément (**C** pour **C**lient, **S** pour **S**erveur et **R** pour un équipement **R**éseau type commutateur, routeur, répéteur). Nous obtenons donc des structures de ce type : $\mathbf{S_cR_rR_s}$, qui symbolise un client simulé connecté à un serveur réel par l'intermédiaire d'un réseau réel.

La première partie va montrer la construction d'un modèle client qui implémente le protocole de communication ModBus/IP ainsi que d'autres fonctionnalités, comme le calcul et la génération des courbes de performance et la réalisation des scénarios de simulation dynamiques, qui vont lui permettre d'être utilisé dans les différentes architectures Co-Simulées étudiées.

La première Co-Simulation est présentée dans le paragraphe 6.3 ; elle est composée du modèle client connecté à un serveur réel via le module HITL de la plateforme de Co-Simulation. Des scénarios dynamiques d'échanges de données entre les deux constituants de cette architecture sont configurés dans le but de montrer l'impact du comportement d'un serveur réel sur la performance du réseau de communication.

Le paragraphe 6.4 propose une comparaison entre les performances des deux architectures Co-Simulées $\mathbf{S_cR_rR_s}$ et $\mathbf{S_cR_rS_s}$. Cette étude a pour but de montrer l'impact du module HITL sur la performance des échanges de communication d'une architecture Co-Simulée.

7.2 Construction du modèle client

Nous avons procédé dans notre étude, pour l'évaluation de performances des architectures Co-Simulées, à la construction d'un modèle de client ModBus/IP auquel sont ajoutés les rôles de planificateur dynamique des requêtes de communication {paragraphe 6.2.1} et calculateur de performance des architectures Co-Simulées {paragraphe 6.2.2}.

La figure 6.1 montre le Process implémenté dans la couche applicative du modèle du client. Ce Process est divisé en quatre niveaux. Le niveau 1 consiste à établir les connexions actives, par l'intermédiaire du bloc 'Connection Establishment', entre le modèle client et les serveurs spécifiés dans les attributs **'Active Connection'** {chapitre 5, paragraphe 5.2.2}. Selon les architectures étudiées, les serveurs peuvent être de type simulés ou réels.

La fonctionnalité de planification dynamique des échanges de communication est une conséquence des données de contrôle envoyées par une application externe SITL vers la simulation en exécution. Cette application de contrôle constitue l'un des composants essentiels du module SITL de notre plateforme de Co-Simulation. La figure 6.1 montre que le niveau 2 du Process applicatif du client permet une acquisition de ces données de contrôle envoyées par l'application SITL.

Le niveau deux est constitué d'une seule étape non forcée intitulée **'Frame_Generation_Core'**. Cette étape contribue au blocage du Process du modèle client en attendant une interruption générée en interne par une fonction OpNet appartenant à l'étape elle-même {Annexe B, process applicatif du modèle client}. Le but de cette interruption revient à réveiller le Process pour le diriger vers l'une des étapes permettant la génération des requêtes protocolaires, en tenant compte du type de la requête et de son temps de génération. Ces deux informations sont spécifiées par l'application de contrôle SITL du modèle.

A l'issue de la réception d'une telle interruption, et en fonction du type de la requête demandée, le Process transitera vers l'une des étapes appartenant au niveau 3. L'implémentation des fonctionnalités de communication du protocole ModBus/IP client est faite dans les étapes associées à ce niveau. Ce dernier est divisé, par conséquent, en quatre étapes distinctes ; deux permettant l'envoi des requêtes de lecture et écriture des informations numériques (i.e. Read n Words, Write n Words) et deux autres permettant l'envoi des requêtes de lecture/écriture des informations logiques (i.e. Read n Bits, Write n Bits) (figure 6.1, $3^{ème}$ niveau).

Les étapes associées au niveau 3 sont de type non forcées, ainsi pareillement à celle existant dans le niveau 2, elles permettent le blocage du process à la fin de leur traitement. Toutefois, le déblocage du Process au niveau 3 est établi par la réception d'une interruption distante venant de la couche TCP et indiquant que la réponse à la requête envoyée a bien été reçue.

A l'issue d'une telle interruption, le Process transite vers le niveau 4 dans lequel le traitement et l'affichage de la réponse envoyée par le serveur seront effectués.

Comme montré sur la figure 6.1, une interaction entre le modèle client et le module HITL est établie aux deux derniers niveaux du Process dans le but de permettre les échanges des requêtes et des réponses entre le logiciel de simulation et le monde réel. Cette interaction aura lieu lorsque le serveur ModBus/IP est réel.

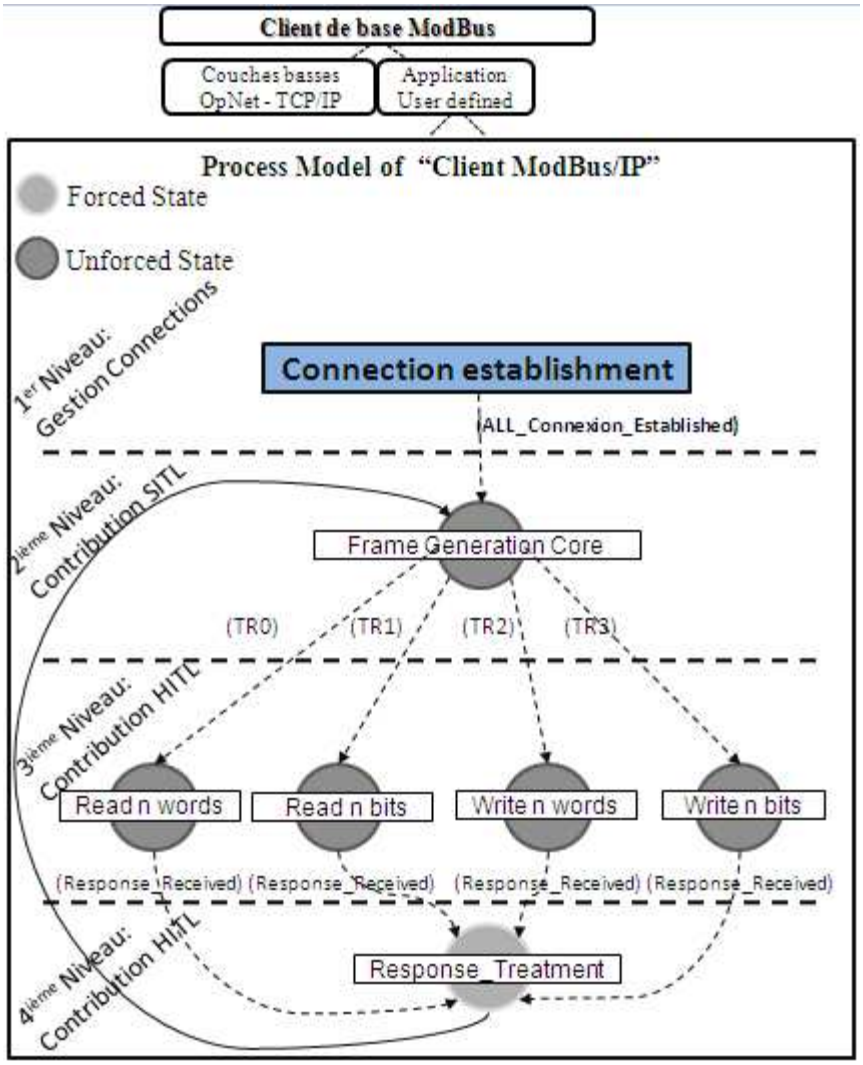


Figure 7.1 Construction du process applicatif du modèle client ModBus/IP

7.2.1 Fonctionnalité de planification dynamique des requêtes de communication

La planification dynamique des requêtes de communication assurée par le modèle client résulte de l'interaction de ce modèle avec le module SITL, plus spécifiquement avec l'application SITL permettant le contrôle du modèle en mode d'exécution de la simulation. Les informations de contrôle du modèle envoyées par l'application SITL via la base de données peuvent être divisées en deux catégories ; la première montre la nature de la requête de communication à envoyer tandis que la deuxième indique la planification horaire de l'envoi de chaque requête.

La figure 6.2 illustre les différentes informations contenues dans l'application de contrôle SITL. La **nature de la requête** de communication est identifiée par les champs 'Code Function', 'Data Number', 'First Adress', 'Data to Write/Read' et 'Value'. 'Code Function' précise la fonctionnalité de la requête de communication (i.e. Lecture/Ecriture Binaire/Numérique). 'Data Number', 'First Adress' précisent le nombre de données à lire ou

écrire et l'emplacement mémoire de la première donnée à manipuler. 'Data to Write/Read' et 'Value' permettent de définir ou de récupérer les valeurs de données à écrire ou lire.

Pour des raisons de calculs de performances précis, on a composé la **planification horaire** des requêtes de communication en trois paramètres :

- Période de génération **'Periodic Value'**: précise la durée d'attente, en millisecondes, du modèle client avant la génération d'une requête de communication. Cette durée est périodique et correspond au temps de cycle pendant lequel le client effectue ses tâches internes.
- Valeur d'incrémentation **'Incremental Value'**: précise la valeur à ajouter à la période de génération dans le cas où le client effectue des tâches plus complexes. En effet, la durée de cycle de traitement d'un client n'est pas fixe, elle dépend de la complexité des tâches effectuées par ce dernier.
- Nombre d'incrémentation **'Incremental Number'** : précise le facteur de multiplication de la valeur d'incrémentation.

Le choix de ces trois paramètres nous permet de réaliser des études détaillées sur la performance des échanges de communication en fonction du délai de traitement du client.

L'information **'Server ID'** est utilisée dans le cas où le client interroge plusieurs serveurs ; elle permet de spécifier le serveur à interroger dans le cas où plusieurs connections actives sont gérées simultanément.

Les deux boutons **'Send'** et **'Send more requests'** permettent la mise à jour de la base de données SITL par les nouvelles requêtes demandées par l'utilisateur. Les valeurs renseignées dans cette base de données seront ensuite récupérés par le modèle client au niveau 2 de son process. La différence entre les deux boutons est que le bouton **'Send'** permet un effacement complet des requêtes existantes dans la base de données pour remettre à jour une nouvelle requête. Tandis que le deuxième bouton rajoute une nouvelle requête à celles existant dans la base.

Ces requêtes ainsi que leurs différentes caractéristiques sont récupérées au niveau de l'étape **'Frame_Generation_Core'** du Process du modèle client.

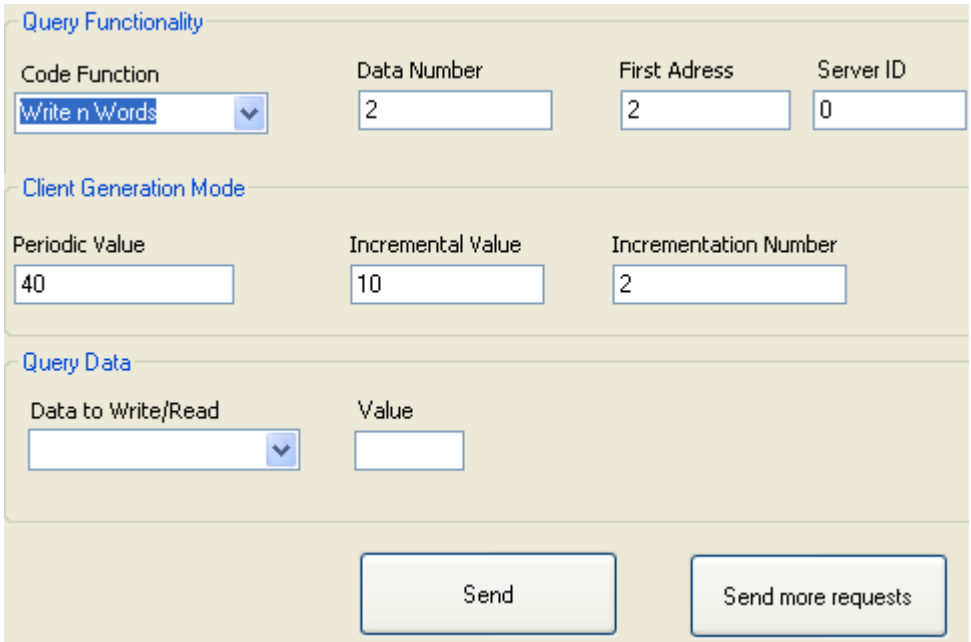


Figure 7.2 Application SITL de contrôle du modèle client ModBus/IP

7.2.2 Calcul des performances réseaux

Le paramètre de performance qu'on a choisi pour l'évaluation des performances des architectures ModBus Co-Simulée est le délai de transaction [Robert et al, 2010]. Ce paramètre représente le temps entre l'envoi d'une requête de communication d'un client et la réception de la réponse envoyée par le serveur.

Le calcul de ce paramètre est effectué au niveau du modèle client, plus spécifiquement, dans le niveau 2 du Process applicatif et en se basant sur des informations mesurées dans les niveaux 3 et 4.

La première information mesurée correspond au temps d'envoi de la requête de communication du client. Cette information est déterminée dans le niveau 3, par l'intermédiaire de la fonction OpNet **'Te = op_sim_time ()'**, exécutée juste après l'envoi de la requête de communication.

La deuxième information correspond au temps de la réception de la réponse serveur. Cette information est calculée dans le niveau 4 en utilisant la fonction OpNet **'Tr = op_sim_time ()'**, implémentée juste après la réception de la réponse du serveur.

Finalement, le paramètre de performance du délai de transaction est calculé dans le niveau 2 du Process en faisant une simple différenciation entre les deux informations mesurées avant de rentrer dans l'étape du niveau 2.

7.3 Co-Simulation d'une architecture $S_cR_rR_s$ en mode multi-scénario dynamique

La première Co-Simulation effectuée dans notre étude a consisté à connecter le modèle client déjà développé à un serveur réel pour construire une architecture mixte $S_cR_rR_s$ (figure 6.4). Le serveur réel choisi est l'automate programmable Premium du constructeur Schneider Electric. La seule étape effectuée au niveau de l'automate a consisté en une configuration de son interface de communication et de son mode d'exécution. La configuration de l'interface de communication a été faite de manière à connecter l'automate sur le même réseau de communication que le modèle client. Le protocole de communication configuré pour cette interface est le ModBus/IP.

Deux modes d'exécution, illustrés sur la figure 6.3, existent pour ce type d'automate: le mode périodique et le mode cyclique. Les deux modes sont caractérisés par trois phases : acquisition des entrées, traitement de programme et activation des sorties [Haffar et al, 2010e]. La différence entre ces deux modes consiste au fait qu'en mode périodique le temps requis pour effectuer un cycle automate est égal à la période définie quelle que soit la durée du traitement du programme. Tandis qu'en mode cyclique ce temps varie en fonction du traitement effectué par l'automate pendant son cycle.

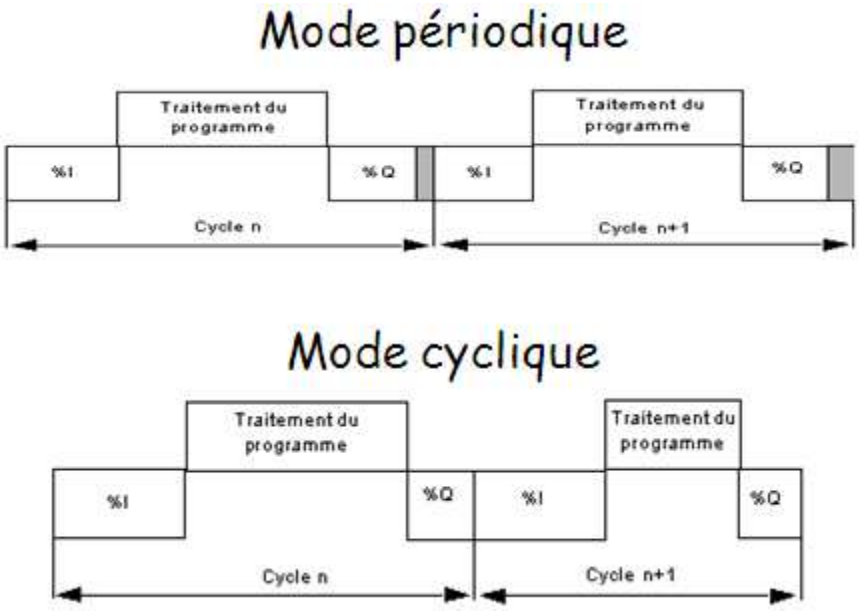


Figure 7.3 Modes d'exécution d'un programme automate programmable

Le mode périodique est celui qui est choisi pour cette Co-Simulation. La période d'exécution a été configurée à la valeur de 100 ms. Le module HITL a été connecté au modèle du client pour lui permettre d'échanger des messages de communication avec le serveur réel.

Pour évaluer l'impact du comportement du serveur sur les performances temporelles de l'architecture de communication étudiée, une simulation d'une durée égale à trois minutes, divisée en trois différents scénarios d'échanges de communication, a été établie par l'intermédiaire des informations de contrôle envoyées du module SITL vers le modèle client.

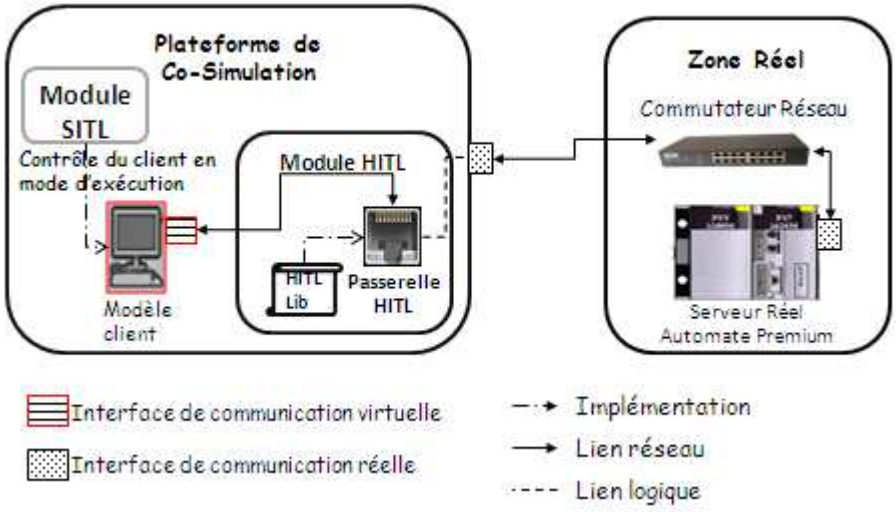


Figure 7.4 Architecture de Co-Simulation $S_cR_rR_s$

Chaque scénario occupe une durée d'une minute du temps de la simulation. Les informations de contrôle liées à la **nature des requêtes** de communication {chapitre 6, paragraphe 6.2.1} ont été identiques pour les trois scénarios de la simulation (lecture un mot à l'adresse 10).

La différence entre ces scénarios concerne la **planification horaire** de ces requêtes {chapitre 6, paragraphe 6.2.1}. Le scénario 1 se caractérise par une période de génération égale à 165 ms, une valeur d'incrémentation égale à 0 ms et un nombre d'incrémentations égal à 0. Le deuxième scénario est configuré avec la même période de génération mais en intégrant une valeur d'incrémentation égale à 10ms et un nombre d'incrémentations égale à 2. Le seul changement effectué dans le dernier scénario concerne la valeur d'incrémentation qui est passée de 2 à 4.

7.3.1 Résultats de performances de la Co-Simulation de l'architecture $S_cR_rR_s$

La figure 6.5 montre l'allure du paramètre de performance **'délai de transaction'** de l'architecture Co-Simulée $S_cR_rR_s$. Cette courbe est générée par l'intermédiaire des statistiques enregistrées dans le modèle client.

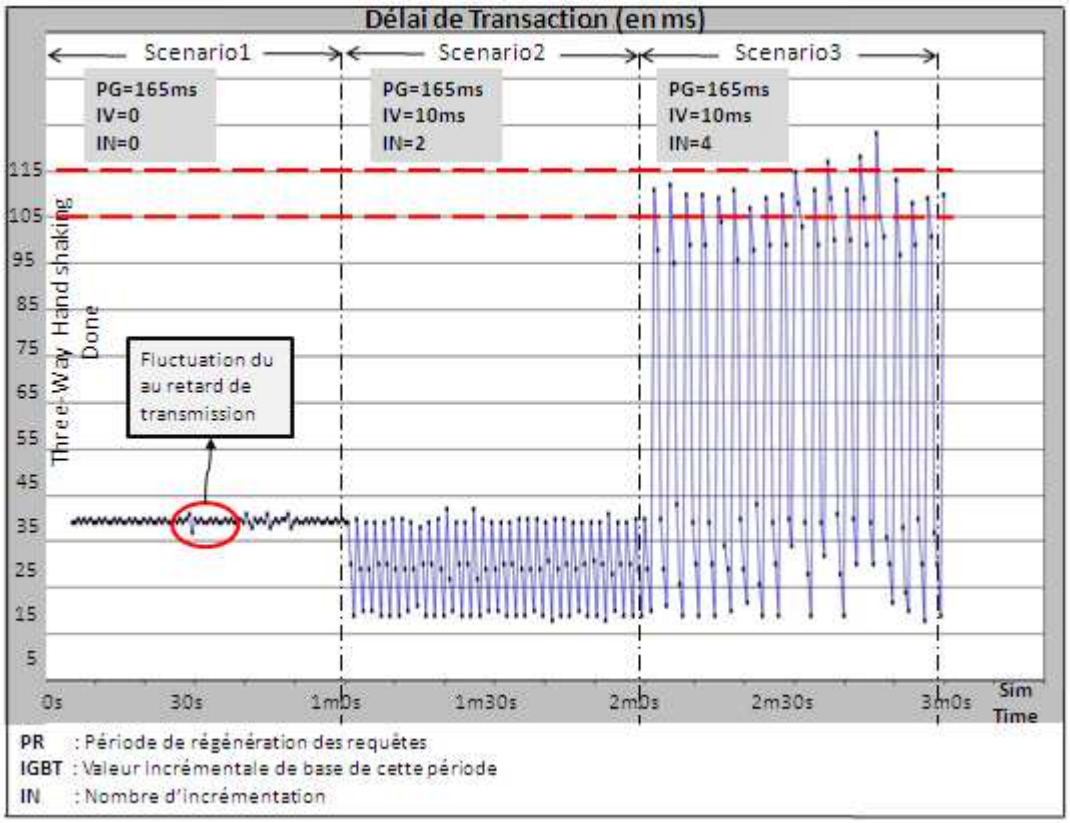


Figure 7.5 Allure des délais de transaction pour les différents scénarios de l'architecture Co-Simulée $S_cR_rR_s$

Le premier point important à déduire de cette courbe est qu'une petite variation au niveau du temps de traitement du client peut contribuer à un changement remarquable au niveau de la performance des messages de communication. Ceci implique que le temps de traitement des équipements (client et/ou serveur) a un impact important sur les performances des messages de communication.

Avant de synthétiser les résultats obtenus, il faut tout d'abord noter que le mécanisme de communication requête/réponse échangé entre le modèle de simulation et l'automate

programmable est complètement synchrone. La synchronisation a lieu après l'envoi des trois communications **'Three Way Hand Shaking'** permettant l'établissement de connexions entre le modèle et l'automate {chapitre 5, paragraphe 5.2.1}. A partir de ce moment, le modèle client démarre par le mécanisme de génération de requête protocolaire (i.e. Niveau 2 du Process du modèle) et le serveur commence un nouveau cycle automate.

Les messages de communication envoyés au cours du scénario 1 sont reçus par le serveur après un cycle complet et 65% du cycle suivant. La valeur de performance obtenue dans ce scénario fluctue autour de **'35 ms'**. Les messages de communication du scénario 2 sont envoyés par le modèle client au rythme de 165 ms, 175 ms et 185 ms ; ainsi ces messages sont reçus par le serveur réel à 65%, 75% et 85% du cycle automate. Par conséquent, l'allure de la courbe de performance de ce scénario varie entre **'35ms'** et **'15ms'** en passant par la valeur **'25ms'**.

Dans le scénario 3 la valeur du paramètre de performance excède la valeur de 100 ms pour certains messages. Plus spécifiquement, les réponses de l'automate correspondant aux messages de communication envoyés par le client à l'instant 195 ms (i.e. à 95 % du cycle automate) sont effectuées après 105 ms.

7.3.2 Analyse de performances de la Co-Simulation de l'architecture $S_cR_rR_s$

Le mode d'exécution de l'automate programmable choisi pour cette étude de performance est décrit dans le chronogramme {chapitre 6, paragraphe 6.3, figure 6.3}. Etant donné que cet automate gère la fonction de communication, un changement au niveau de ce chronogramme aura lieu pour prendre en compte les communications entrantes et sortantes de l'automate (figure 6.6). En effet, les communications entrantes (i.e. les requêtes envoyées par le modèle client ModBus/IP) sont « bufférisées » en mode évènementiel lors de leur arrivée à ce dernier et les réponses associées à ces requêtes sont envoyées à la fin du cycle automate (figure 6.6).

La flèche **'Req A'** (figure 6.6) indique une requête envoyée par le modèle client et reçu par l'automate programmable avant la fin de son cycle. Le temps séparant l'envoi de la requête par le modèle client et sa réception par l'automate correspond à des retards variés (latence des commutateurs, propagation des messages, retard au niveau des modules de la plateforme, etc). La réponse à cette requête **'Rep A'** est transmise au modèle client à la fin du cycle automate. **'Req A'** et **'Rep A'** correspondent aux communications échangées entre le modèle client et l'automate dans les scénarios 1 et 2. En effet, une fois la synchronisation établie entre le modèle client et l'automate, les messages de communication du scénario 1 sont envoyés périodiquement vers l'automate après un laps de temps égal à 165 ms et sont reçues ainsi par ce dernier après un cycle complet et 65 ms du cycle suivant auxquels s'ajoutent divers retards (figure 6.6). En prenant en compte ces différents délais, les messages de communication du scénario 1 sont bufférisés par l'automate à 65% de son cycle ; ce dernier constitue leurs réponses à la fin du cycle. Par conséquent, le modèle client reçoit les réponses à ces requêtes après 35% du cycle automate d'où la valeur affectée à la performance du délai de transaction qui fluctue aux alentours de 35 ms. L'allure de performances associée au scénario 2 confirme cette analyse. En effet, la variation du délai de traitement du client dans ce scénario entre 165 ms, 175 ms et 185 ms (i.e. réception par l'automate à 65%, 75% et 85% du cycle), a causé une variation du délai de transaction entre 35 ms et 15 ms.

La flèche **'Req B'** (figure 6.6) indique une requête envoyée par le modèle client juste avant la fin du cycle automate et reçu par ce dernier juste après le démarrage d'un nouveau cycle ; la réponse **'Rep B'** associée à cette requête est transmise au modèle après un cycle automate complet. Les messages de communication du scénario 3 envoyés par le modèle client avec une période égale à 195 ms rentrent bien dans le cas de figure des **'Req B'** et **'Rep B'**. En effet, vu la synchronisation ces messages sont envoyés par le modèle client après un cycle complet et 95 % d'après de l'automate. Les retards variés montrés sur la figure 6.6 pour la flèche **'Rep B'** causent l'arrivée de ces messages de communication après le démarrage d'un nouveau cycle. Par conséquent les performances associées à ces messages ont varié comme le montre la figure 6.5 aux alentours de 105 ms au lieu de varier aux alentours de 5 ms.

Cette analyse nous permet d'une part de montrer l'intérêt du modèle client ModBus/IP au niveau des Co-Simulations et d'autre part de valider les résultats de performances délivrés par notre plateforme de Co-Simulation sur des architectures mixtes composées d'équipements réels et virtuels.

Toutefois comme on vient de voir dans cette analyse, les retards variés peuvent avoir dans certains cas des impacts sévères sur les performances. La Co-Simulation effectuée dans le paragraphe suivant vise à montrer l'origine de ces retards.

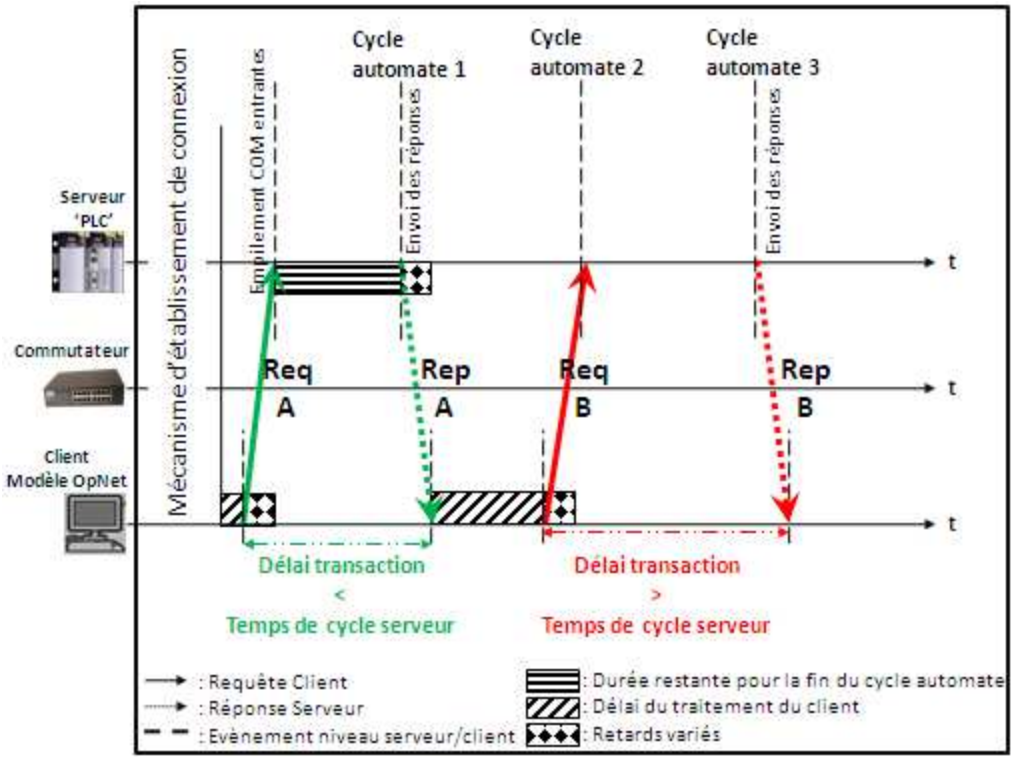


Figure 7.6 Mécanisme des échanges de communication entre les constituants de l'architecture Co-Simulées $S_cR_rR_s$

7.4 Co-Simulation des architectures $S_cR_rR_s$ et $S_cR_rS_s$ et analyse des résultats de performance

La Co-Simulation effectuée dans ce paragraphe a pour but d'identifier les retards imposés par les modules de la plateforme et d'analyser l'impact de ces retards au niveau des performances des architectures Co-Simulées. Cette Co-Simulation va présenter une évaluation et une comparaison des performance entre deux architectures de communication mixtes montrées sur la figure 6.7.

La première architecture est de type $\mathbf{S_cR_rR_s}$, elle est composée d'un client virtuel connecté à un serveur réel via le module HITL de la plateforme de Co-Simulation et des équipements réseaux réels. Le serveur réel choisi dans cette étude est le simulateur Unity Pro des automates programmable du constructeur Schneider Electric.

La deuxième architecture est de type $\mathbf{S_cR_rS_s}$, cela signifie que le serveur est ici simulé. La connexion entre le modèle client et le modèle serveur de cette architecture est réalisée par l'intermédiaire des modules HITL des deux plateformes de Co-Simulation.

Une seule Co-Simulation se basant sur un mode d'exécution multi-scénario a été réalisée afin d'évaluer les performances des deux architectures. En effet, le modèle client est utilisé en commun pour les deux architectures {chapitre 6, paragraphe 6.2}. Le but essentiel de cette Co-Simulation est de comparer l'impact du modèle serveur et d'un serveur réel sur les performances de l'architecture de communication afin d'identifier les retards imposés pour le transfert des communications entre les deux mondes réels et virtuel (i.e. retard imposé par le module HITL de la plateforme de Co-Simulation).

Etant donné qu'aucune notion de temps de cycle n'est implémenté dans notre modèle serveur montré et validé dans le {chapitre 5, paragraphe 5.5.1}, nous avons procédé dans cette Co-Simulation à configurer le simulateur d'automate en mode cyclique. Aucun programme n'a été implémenté dans le simulateur d'automate afin de lui permettre de jouer le rôle d'un simple serveur de communication ModBus/IP. Cette configuration faite au niveau de l'automate nous permet de faire une comparaison précise entre l'impact d'un équipement serveur réel et d'un serveur virtuel sur les performances d'une architecture de communication.

Une simulation d'une durée de deux minutes divisée en deux scénarios d'échanges de données configurés par le module SITL a été exécutée. Les informations de contrôle envoyée par l'application SITL au modèle client sont identiques à celles de la Co-Simulation précédente du {chapitre 6, paragraphe 6.3}. La même nature des requêtes de communication (i.e. écriture information numérique à l'adresse 10) ainsi que la même planification horaire des requêtes (périodique d'une seconde) ont été configurés pour les deux scénarios d'échanges de communication. La seule différence entre ces deux scénarios était au niveau de l'information de contrôle 'Server ID' permettant de différencier entre le serveur réel et le modèle existant dans la deuxième plateforme de Co-Simulation.

Le scénario 1 a occupé une durée d'une minute et a permis l'échange entre le modèle client et le simulateur d'automate programmable. Le scénario 2 a occupé la deuxième minute de la simulation et a permis l'échange de communication entre le modèle client et le modèle serveur de la deuxième machine de Co-Simulation.

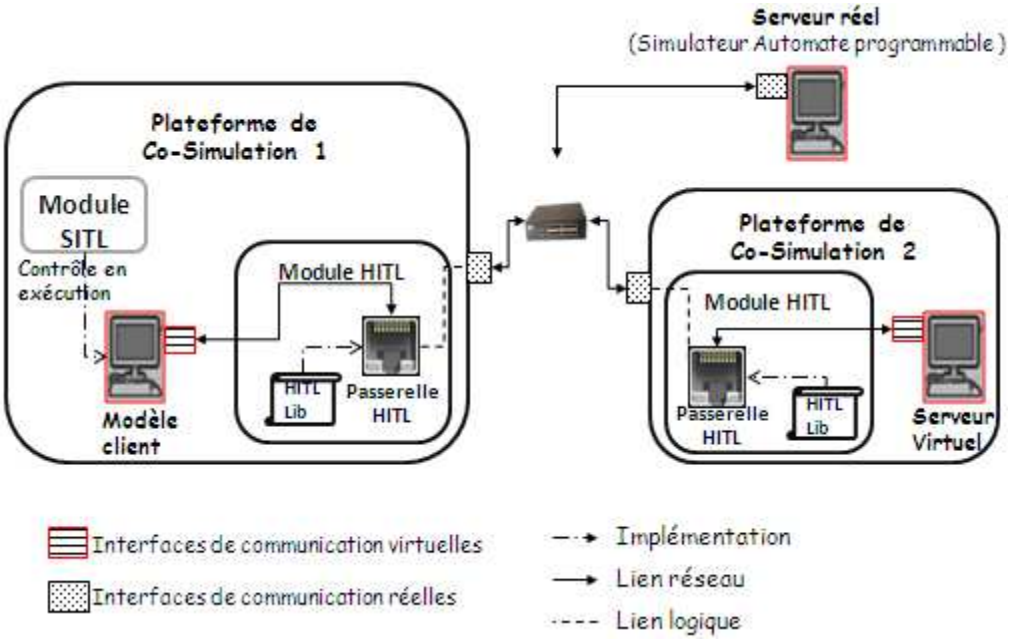


Figure 7.7 Constituants essentiels des architectures $S_cR_rS_s$ et $S_cR_rR_s$

La figure 6.8 montre les moyennes des deux courbes de performances associées à chaque scénario de la Co-Simulation effectuée.

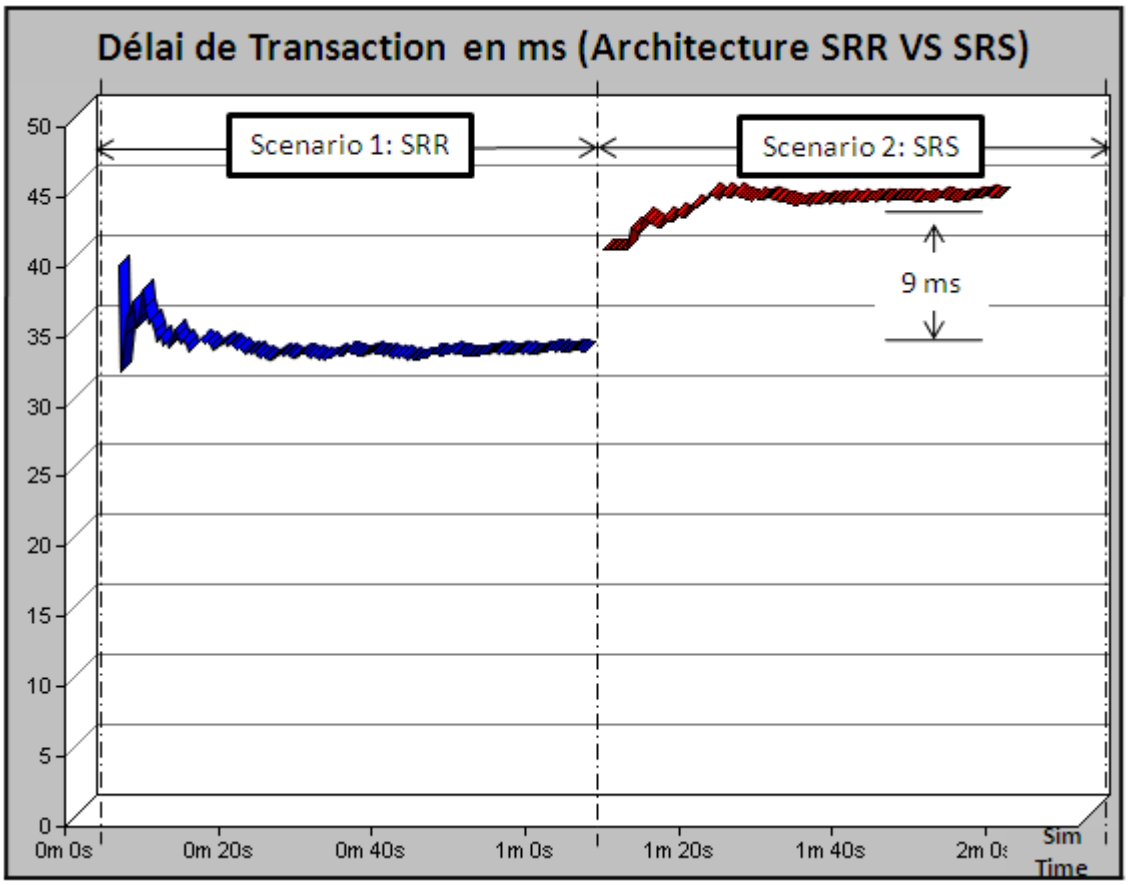


Figure 7.8 Performance de la Co-Simulation des architectures $S_cR_rR_s$ et $S_cR_rS_s$

Pour le premier scénario, le délai de transaction a varié aux alentours de la valeur 33 ms tandis que dans le deuxième scénario la valeur de performance a varié aux alentours de la valeur 42 ms (figure 6.8).

7.4.1 Analyse de performances de la Co-Simulation des architectures $S_cR_rR_s$ et $S_cR_rS_s$

L'analyse de ces résultats de performance est montrée par les deux diagrammes temporels associés à chaque architecture mixte (figure 6.9). Ces diagrammes montrent les différents paramètres de retard constituants chacune des deux courbes de performances. La différence entre les deux diagrammes est montrée en pointillés sur la figure 6.9. Cette partie est associée à l'architecture SRS, elle symbolise le retard ajouté par le module HITL de la plateforme de Co-Simulation 2. Plus spécifiquement, ce retard contient le temps de routage de la trame du composant HITL vers la passerelle HITL, filtrage de la trame par la passerelle et conversion du format de la trame entre le réel et le simulé {chapitre 4, paragraphe 4.5.2}.

Les autres retards (a, b, c, d et e) sont les mêmes pour les deux architectures du fait que ces deux architectures englobent les mêmes constituants.

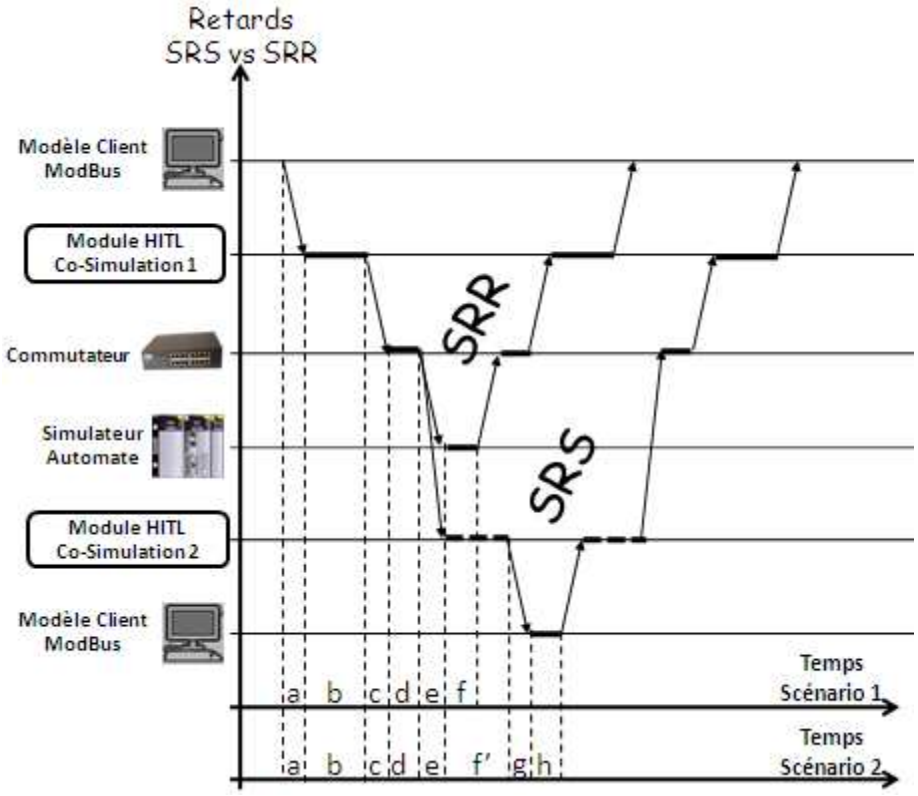


Figure 7.9 Diagrammes temporels des échanges de communication des deux architectures Co-Simulées SRR & SRS

Le paramètre '**a**' présente le délai d'encapsulation/décapsulation de la trame au niveau des couches réseau du modèle client ainsi que le délai de propagation de la requête depuis le modèle vers la passerelle HITL. Le paramètre '**b**' symbolise le retard dû au traitement du message de communication par le module HITL de la plateforme de Co-Simulation 1, ce retard est commun pour les deux architectures. Le paramètre '**c**' englobe les délais de transmission de la trame réelle par l'intermédiaire de l'interface de communication de la

machine de Co-Simulation et le délai de propagation de la trame de l'interface de communication de la machine de Co-Simulation 1 vers celui du commutateur réseau fait aussi partie de ce paramètre. Le paramètre '**d**' définit le délai de latence au niveau du commutateur réseau réel. Le paramètre '**e**' indique le délai de transmission et de propagation depuis l'interface de communication du commutateur vers l'interface de communication du serveur (réel ou virtuel). Finalement, les deux paramètres '**f**' et '**h**' correspondent aux délais de réception, décapsulation et traitement du modèle serveur.

En conséquence, la différence entre les deux diagrammes temporels persiste au niveau des deux paramètres '**f**' et '**g**'. Le paramètre '**g**' présente le délai de propagation du message de communication de la passerelle HITL vers le serveur virtuel. Ce paramètre peut être négligé vu que le lien reliant la passerelle HITL au modèle client est de type 100Mbits/sec ; ainsi pour une trame ModBus/IP de taille égale 8*((91 + x) + (90)) ou x est le nombre de données en octets, le délai de propagation sera dans l'ordre de 0.014 ms ce qui est négligeable compte tenu des performances obtenues.

Le paramètre '**f**' correspond au retard imposé par le module HITL. Ce paramètre est mesuré par le biais des statistiques OpNet associées au module HITL et générées à la fin de chaque Co-Simulation. La figure 6.10 montre l'allure des retards des modules HITL pour les deux architectures Co-Simulées. Nous pouvons remarquer que le premier scénario contient une seule allure du fait de l'utilisation d'une seule plateforme de Co-Simulation du modèle client, tandis qu'une deuxième allure qui correspond au délai du module HITL de la plateforme de Co-Simulation du modèle serveur est rajoutée pour le deuxième scénario. Ces deux allures varient en moyenne aux alentours de la valeur 9 ms ce qui explique la différence obtenue entre les deux courbes de performance des deux architectures SRR et SRS (figure 6.8).

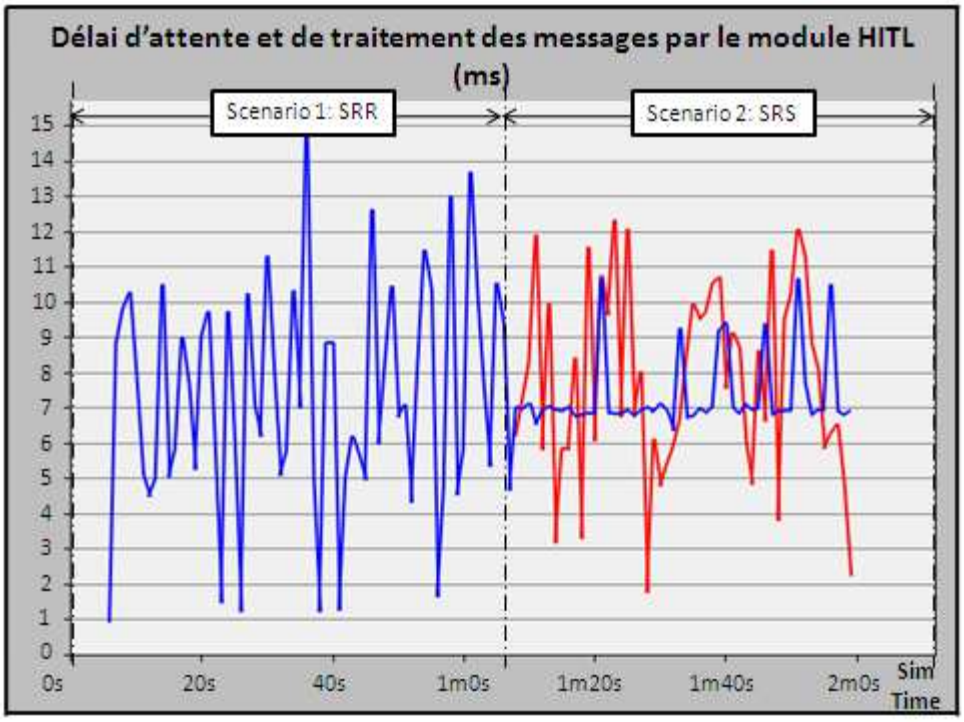


Figure 7.10 Retards des modules HITL des deux plateformes de Co-Simulation

7.4.2 Discussion autour de l'impact du retard HITL sur les performances d'une Co-Simulation IEC 61850

Nous avons choisi dans cette dernière Co-Simulation de montrer la courbe réelle du retard des modules HITL afin de préciser que la valeur de ce retard n'est pas fixe et peut atteindre dans certains cas les 12 ms (figure 6.10).

Ce retard peut avoir des impacts négatifs sur les Co-Simulations réalisées lors de la phase de vérification dynamique d'un SDC IEC 61850. En effet, cette phase consiste à tester les automatismes distribués entre des SDP IEC 61850 réels et d'autres virtuels substituant les SDP inexistant lors de cette phase de vérification {chapitre 3, paragraphe 3.5.3}. La vérification de ces automatismes distribués est liée aux performances des signaux sécuritaires échangés entre les équipements réels et les modèles via le paquetage HITL de la plateforme de Co-Simulation {chapitre 3, paragraphe 3.5.4}.

Etant donné que les signaux sécuritaires doivent respecter des exigences temporelles imposées par la norme de communication IEC 61850 {chapitre 3, paragraphe 3.3.4}, les retards ajoutés par les modules HITL, même qu'ils sont de l'ordre de 10 ms, peuvent causer la non viabilité {chapitre 3, paragraphe 3.4} des signaux sécuritaires vis-à-vis des automatismes distribués utilisant ces signaux. Ceci aboutira à un dysfonctionnement de ces automatismes distribués lors de la phase de vérification non pas à cause d'une défaillance du système réel mais à cause des retards introduits par les modules de la plateforme.

Par conséquent, pour certains types d'utilisation, notre plateforme de Co-Simulation possède des limites qui doivent être prises en compte lors des phases d'évaluation et de vérification de performance d'un SDC IEC 61850.

7.5 Conclusion

Ce chapitre a montré l'utilisation de notre plateforme de Co-Simulation avec pour objectif l'évaluation de performance des architectures mixtes comportant des équipements réels et des équipements simulés.

L'allure de performance obtenue lors de la Co-Simulation de l'architecture SRR nous a permis d'une part de montrer l'intérêt des modules HITL et SITL de la plateforme et d'autre part de valider les résultats fournis par cette dernière. Le troisième scénario de communication de cette Co-Simulation nous a permis d'identifier qu'il y a des retards qui sont imposés par les modules de la plateforme et qui peuvent dans certains cas avoir des impacts importants sur les performances de cette dernière. La deuxième Co-Simulation a été effectuée avec pour objectif d'évaluer les retards du module HITL sur les performances d'une architecture Co-Simulée. Cette Co-Simulation a réuni deux architectures : la première de type SRR et la deuxième de type SRS. Une comparaison entre les performances de ces deux architectures nous a incité à justifier les retards imposés par les modules HITL de la plateforme. Ces retards doivent être pris en compte lors des phases de vérification et d'évaluation des performances des signaux sécuritaires d'un SDC IEC 61850 et peuvent dans certains cas induire des limitations sur l'utilisation de cette plateforme de Co-Simulation.

Chapitre 8

Modélisation du standard de communication IEC 61850

8.1 Introduction

Le but essentiel de cette plateforme consiste à réaliser des tests avancés lors des phases de conception et de vérification d'un nouveau SDC IEC 61850. Ces phases exigent la présence des modèles de communication conformes à la norme de manière à ce qu'ils soient capables de se connecter aux vrais équipements IEC 61850 présents lors de ces phases {chapitre 3, paragraphe 3.5.4}.

Par conséquent, à la différence de la plupart des études que l'on trouve dans la littérature [Ali et al, 2008], [Sidhu et al, 2007] et [Thomas et al, 2010], notre étude a pour objectif l'implémentation de vrais formats de paquets appartenant aux services de communication du standard IEC 61850.

L'implémentation des services de communication au niveau de la couche applicative des modèles de communication doit être au préalable validée avant l'utilisation de la plateforme de Co-Simulation. Ce chapitre présente les premières étapes de modélisation du standard IEC 61850 dans le logiciel de simulation OpNet Modeler et les tests effectués pour valider la partie modélisée.

Le paragraphe 7.2 montre l'environnement de travail pour le standard IEC 61850.

Le paragraphe 7.3 fait état de l'aspect informatique des différents fichiers du fournisseur de code choisi et explique comment s'adaptent ces fichiers en fonction de l'environnement de notre système d'exploitation.

Le paragraphe 7.4 décrit les premières implémentations du standard de communication IEC 61850 dans la plateforme de Co-Simulation. Les résultats de cette implémentation indiqués à la fin permettent de valider la première phase de modélisation du protocole IEC 61850.

8.2 Environnement de travail IEC 61850

À la différence du protocole de communication ModBus, le standard IEC 61850 n'est pas considéré comme un protocole libre de droit 'Open Source'. Par conséquent, les sources code de ce standard de communication ne sont pas accessibles en téléchargement gratuit. Ces sources sont vendues par des entreprises mondialement reconnues dans le domaine des sources protocolaires.

Suite à l'obtention d'un financement OSEO pour notre étude de recherche, notre entreprise s'est mise en collaboration avec l'entreprise américaine **TMW** '**T**riangle **M**icro**W**orks' pour l'achat de sources protocolaires IEC 61850. TMW est l'une des entreprises les plus connues dans le monde des protocoles de communication.

Les **sources code protocolaires** vont servir de point de départ pour développer les applications IEC 61850. Ce développement consiste à implémenter les fonctionnalités de communication de ce standard au niveau des couches applicatives des produits cibles.

Dans notre cas, les produits cibles constituent les modèles de communication de la plateforme de Co-Simulation. À la suite de cette implémentation, la section de communication d'une unité de protection sera modélisée dans la plateforme de Co-Simulation.

Les **sources code protocolaires,** délivrées par la société **TMW,** permettent de mettre à disposition les services de communication contenus dans la couche **ACSI,** décrits dans {chapitre 3, paragraphe 3.2.4}. Cette abstraction permet au développeur des applications 61850 (i.e. couches applicatives) d'avoir accès à ces services de communication sans même savoir, d'une part, comment ceux-ci sont mappés aux couches basses et d'autre part, quel est le format du paquet de communication constituant ces services.

Mises à part les sources codes permettant la programmation des applications IEC 61850, ce fournisseur met à disposition les deux outils essentiels :

- **IEC 61850 Test Harness**
- **IEDScout**

L'outil **'IEC 61850 Test Harness'** a pour rôle principal de fournir une solution de configuration des applications IEC 61850. Cet outil permet de même la génération des fichiers ICD permettant la description des fonctionnalités contenues dans un serveur IEC 61850.

De même, cet éditeur permet la configuration des applications IEC 61850 en générant leurs fichiers de configuration CID.

Cet éditeur permet la génération du fichier CID en utilisant soit la méthode décrite dans le standard IEC 61850 (i.e. découpage du fichier global SCD) soit en partant des fichiers de capacité ICD.

Dans ce dernier cas, cet éditeur génère un fichier intermédiaire appelé IID contenant seulement les messages Goose publiés par l'application. Pour la configuration de l'aspect

d'abonnement des Goose, une réintégration des fichiers IID des autres applications sera nécessaire. A la suite de cette réintégration, le fichier CID de l'application sera créé.

La figure 7.1 illustre le principe de cette configuration pour une architecture simple composée de deux applications serveur IEC 61850.

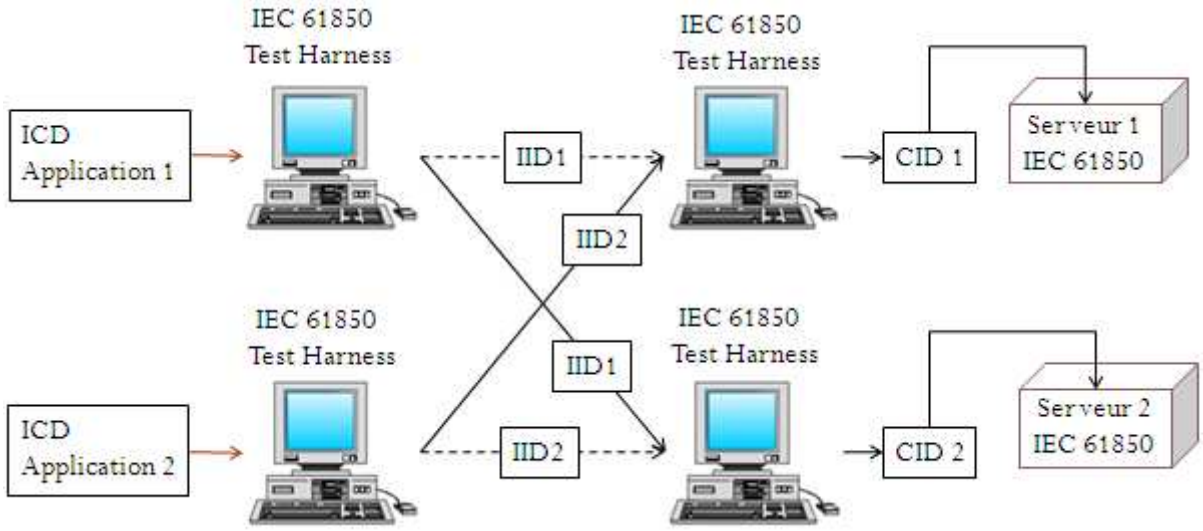


Figure 8.1 Configuration des applications IEC 61850 en passant par les fichiers IID

Les fichiers de configuration générés par cet outil sont tous conformes à la partie 6 du standard IEC 61850. Cet outil pourra être perçu comme l'éditeur SCL fourni par TMW.

L'IEDScout est un outil universel jouant le rôle d'un client IEC 61850. Il est développé par la société Omicron et livré par la société TMW. Son but est de tester le développement des applications serveur IEC 61850. Il permet aussi d'assurer une connexion à plusieurs applications serveur simultanément.

Les tests sont accomplis via l'envoi d'un ensemble de services de communication vers les applications serveurs IEC 61850 (i.e. applications développées).

Les services de communication assurés par cet outil sont:

- Auto-Découverte de l'implémentation objet des équipements 61850. Le but de ce service revient à faciliter le développement des systèmes de supervision. Il pourra aussi être utilisé (i.e. dans le cas de l'IEDScout) pour vérifier le contenu objet d'un équipement IEC 61850 et ce, pour le comparer à son fichier de configuration CID

- Echange des services de communication MMS avec les serveurs dans le but de tester les communications conventionnelles de lecture/écriture des objets appartenant au serveur

- Capacité à recevoir les rapports de données. Ceci permet de tester les services 'Reporting' intégrés dans l'équipement serveur IEC 61850

- Possibilité de Souscription/Émission des messages Goose {chapitre 3, paragraphe 3.2.4} dans le but de tester ce type d'échange de communication. Ces messages sont généralement échangés entre les serveurs. Cette fonctionnalité permet de tester les Goose implémentés dans les applications serveur IEC 61850.

Les deux outils sont livrés par la société TMW afin de configurer et de tester toutes les applications serveur IEC 61850 développées au travers des sources code protocolaires de cette société.

8.3 Modélisation du standard IEC 61850 dans l'environnement de la plateforme de Co-Simulation

Les études de simulation montrées dans {chapitre 3, paragraphe 3.4.2.3} avaient pour objectif d'évaluer les performances des architectures de communication IEC 61850, pendant la phase de conception, afin de choisir celle la mieux adaptée aux critères de performances exigées par la norme {chapitre 3, paragraphe 3.6}. Cette phase n'exige pas une implémentation réelle du standard de communication du fait que les simulations doivent se réaliser dans un environnement complètement simulé. Les messages de communication simulés doivent posséder à ce stade les mêmes caractéristiques que les vrais messages IEC 61850 mais sans en avoir nécessairement le même format. Parmi les caractéristiques essentielles des messages de communication, on cite les suivants :

- Taille des messages
- Profil de communication utilisé par les messages application 61850 (e.g. 'Ethernet pour les Goose' et 'TCP/IP pour les MMS'
- Niveau de priorité

La littérature expliquée dans {chapitre 3, paragraphe 3.4.2.3} montre des études de simulation pour l'évaluation de performance des systèmes de communications IEC 61850. Ces simulations utilisent des modèles possédant les mêmes profils de communication que ceux de la norme IEC 61850 (profil Application-Liaison 'Ethernet', profil Application-Transport 'TCP', profil Application-Transport 'UDP'). En plus, les caractéristiques du protocole de communication implémenté au niveau applicatif de ces modèles ressemblent à ceux spécifiés dans la norme (i.e. même taille des messages applicatifs, même niveau de priorité, etc..). Néanmoins, du fait que ces études visent à évaluer les performances au niveau des phases de conception de l'architecture de communication, aucune de ces études montrent une implémentation réelle des vrais formats de message du protocole IEC 61850 :

- Quel fournisseur des sources protocolaires a été choisi ?
- Quelle méthodologie est utilisée pour l'implémentation des sources codes du standard ?
- Comment sont validés les services de communication IEC 61850 implémentés dans les modèles?

La modélisation protocolaire faite dans la littérature demeure logique du fait que les simulations faites jusqu'à ce jour ne prennent en compte aucune évaluation de performance d'un système de communication IEC 61850 en établissant des connexions entre les modèles et des équipements réels.

Notre problématique de recherche impose à établir des communications entre les modèles et les vrais systèmes de protection IEC 61850 {chapitre 3, paragraphe 3.5}. En effet, les modèles conçus dans notre étude vont être utilisés non seulement au niveau de la conception d'un SDC IEC 61850 mais aussi lors de la phase de vérification de ce SDC. Par conséquent, les deux parties de cet échange doivent être conformes à l'implémentation du standard de communication IEC 61850. Pour cette raison et à la différence de toutes les études existantes dans la littérature nous avons opté pour l'implémentation du vrai format de messages du standard IEC 61850 dans nos modèles de simulation.

La construction de ces modèles IEC 61850 consiste à implémenter les différents services et fonctionnalités de communication du standard dans les couches applicatives de ces modèles et de tester la conformité de cette implémentation. Cette étape a été réalisée en ayant recours aux fonctions appartenant à des librairies livrées par le fournisseur TMW qui doivent être, par conséquent, implémentées dans l'environnement de la Co-Simulation (figure 7.2, étape 3).

La génération de ces librairies doit passer tout d'abord par l'identification des différents fichiers informatiques constituant l'environnement de développement de TMW (figure 7.2, étape 1). Ceux les plus essentiels sont les fichiers de compilation et les fichiers sources protocolaires {Annexe D}.

Différentes phases d'adaptation ont été appliquées à ces fichiers pour les rendre compatibles avec l'environnement de travail de la plateforme de Co-Simulation (figure 7.2, étape 2).

Plus spécifiquement, la première adaptation a été effectuée au niveau des fichiers de compilation afin de les rendre compatible avec l'environnement de programmation de la plateforme de Co-Simulation {Annexe D}. Ceci a permis la réussite de l'exécution des fichiers de compilation par l'intermédiaire de leurs utilitaires et la génération des librairies associées aux fichiers sources de TMW. Ces librairies {Annexe D} contiennent les fonctions informatiques permettant la programmation de la couche applicative des modèles ou des équipements, conformément au standard IEC 61850.

Toutefois, les librairies obtenues après la première phase de compilation n'ont pas abouti à la réussite de la tâche de modélisation du standard IEC 61850. En effet, après un débogage informatique du problème, on a constaté que les fonctions informatiques appartenant aux fichiers sources TMW ne sont pas compatible avec l'environnement de communication des modèles de simulation OpNet. Ceci nous a amené à ajouter une nouvelle phase d'adaptation au niveau des fichiers sources **'.C'** de TMW pour les rendre compatible avec le stack de communication des modèles OpNet (partie soulignée de l'étape 2 de la figure 7.2). Les points modifiés dans ces fichiers sources vont faire l'objet du paragraphe suivant.

Cette adaptation était l'un des points incontournables pour la modélisation du standard IEC 61850. A l'issue de cette modification, une régénération des librairies, par l'intermédiaire des fichiers de compilation modifiés, a été effectuée afin de prendre en compte les changements des fichiers sources de TMW. L'utilisation des fonctions appartenant à ces librairies a permis de réussir la première phase de modélisation du standard IEC 61850 dans la plateforme de Co-Simulation. Les détails de cette modélisation sont indiqués dans les paragraphes suivants.

La figure 7.2 illustre les différentes étapes permettant la génération des librairies de fonctions adaptées à l'environnement de la plateforme de Co-Simulation.

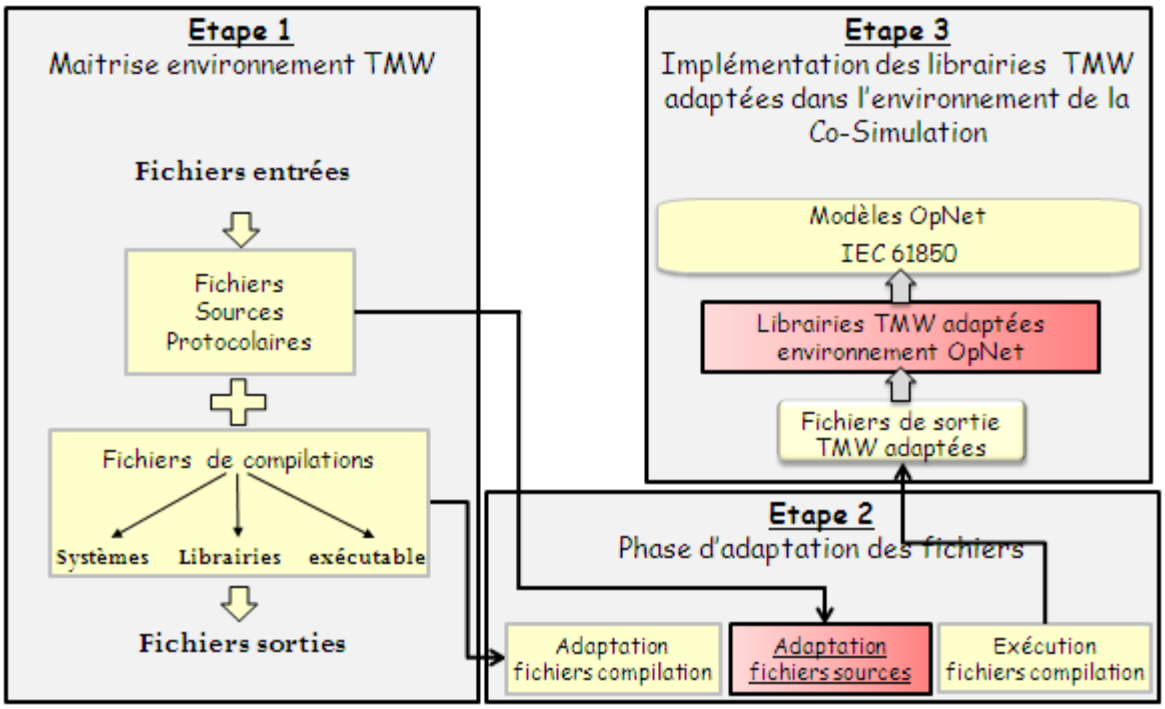


Figure 8.2 Etape de génération des librairies de fonctions IEC 61850-TMW pour l'environnement de communication de la plateforme de Co-Simulation

8.3.1 Adaptation effectuée au niveau des fichiers sources TMW

Le standard IEC 61850 implémente plusieurs profils dont chacun est constitué de plusieurs services de communication {chapitre 3, paragraphe 3.2.4}. La première modification effectuée au niveau des fichiers sources TMW a consisté à rendre ces derniers compatibles avec le stack TCP/IP d'OpNet et ce, afin de réussir l'implémentation des services de communication MMS-TCP/IP dans les modèles OpNet.

En effet, les fichiers initiaux de TMW ont pour but d'une part la construction des messages applicatifs conformément à la norme et d'autre part l'envoi et la réception de ces messages via le stack TCP/IP d'un système d'exploitation Windows ou Linux.

Le stack de communication des modèles OpNet diffère de celui de l'interface de communication de la vraie machine implémentant les systèmes d'exploitation standard (i.e. Windows ou Linux). Les stack de communications standards ont recours à une librairie de fonction système intitulée 'Winsock.h' dans le but de l'envoi et la réception des communications TCP/IP. La modification des sources TMW a ainsi consisté au remplacement de toutes les fonctions de cette librairie par celles correspondant à OpNet et permettant l'échange de communication TCP/IP via le stack des modèles de simulation.

Après un balayage des fichiers sources TMW, nous avons pu localiser l'emplacement de ces fonctions d'interfaçage avec le stack TCP/IP. Ces fonctions ont été trouvées dans le fichier source TMW intitulé 'RFC1006.c'. Ce fichier après compilation génère une librairie de fonction TMW intitulée '1006.lib'.

L'adaptation de ce fichier est divisée en deux parties. La première partie représentée sur la figure 7.3 montre les modifications effectuées pour rendre compatible les fonctions **d'envoi** des paquets MMS-IEC61850 via le stack TCP/IP des modèles OpNet. La deuxième partie, illustrée sur la figure 7.6, représente les modifications dans le sens de la **réception** de ces paquets.

Pour l'envoi des paquets, deux étapes ont été requises. La première a consisté au remplacement du paquet applicatif MMS-IEC61850 construit par les fonctions TMW par un équivalent créé dans l'environnement OpNet. Le paquet OpNet, intitulée 'Response1' (figure 7.3), a été construit d'une manière non formatée dans le but d'incruster la suite des octets du paquet applicatif de TMW. Les fonctions qui ont permis ce remplacement sont montrées dans le premier rectangle de la figure 7.3. La deuxième étape de l'adaptation dans le sens d'envoi des paquets a consisté à remplacer la fonction 'send' de la librairie 'Winsock.h', par la fonction 'tcp_data_send' équivalente dans la librairie **'tcp_api_v3.h'** d'OpNet {chapitre 5, paragraphe 5.2.1}.

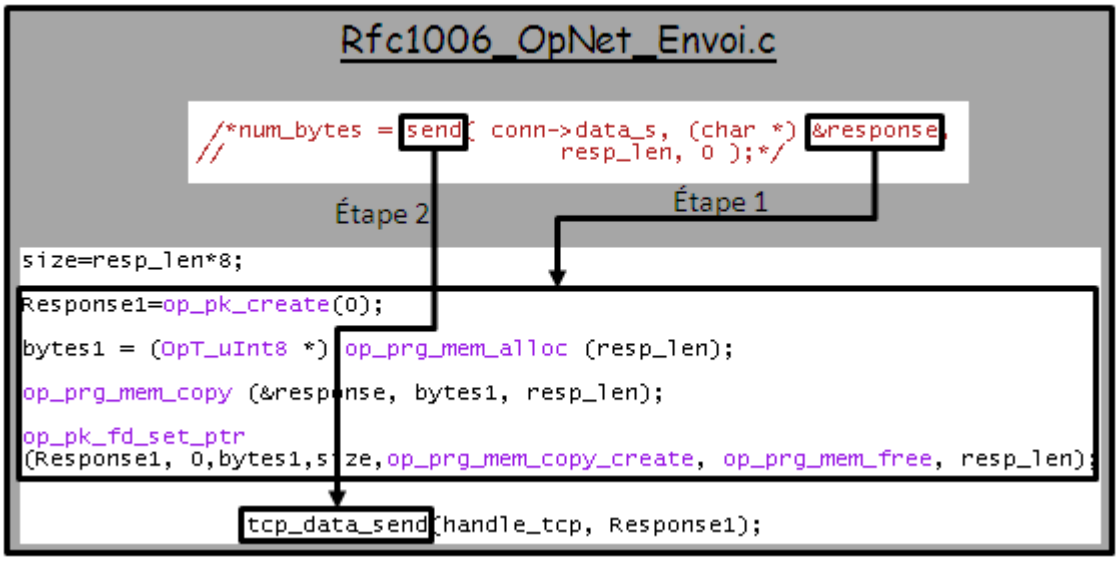


Figure 8.3 Modification fichier 'rfc1006.c' pour l'envoi des paquets MMS/IEC61850 via OpNet

Ce qui est important à souligner au niveau dans la deuxième partie de cette adaptation c'est que lors d'une réception d'un paquet applicatif MMS, le fichier source initial Rfc1006.c procède au rangement des champs de ce paquet dans un tableau de caractères alloués dans la mémoire de l'application IEC 61850 et intitulé **'bytes'**. Tous les autres fichiers sources de TMW effectuent leur traitement sur ce tableau alloué sous le nom de **'bytes'**. Afin de ne pas effectuer des changements au niveau de ce traitement, on a procédé à un unique changement pour l'adaptation dans le sens de la réception.

Ce changement a consisté juste à ce que lors de la réception d'un paquet MMS-TCP/IP par les modèles de simulation, ce paquet sera extrait de la couche TCP du modèle d'OpNet en utilisant les deux fonctions OpNet **'tcp_receive_command_send'** et **'op_pk_get'** de la

librairie **'tcp_api_v3.h'** (figure 7.4). Ces fonctions sont équivalentes à la fonction **'recv'** de **'Winsock.h'**.

La figure 7.4 montre que le paquet applicatif extrait par ces fonctions sera rangé dans le même emplacement mémoire alloué par les fonctions initiales de TMW **'bytes'**, et ce pour ne pas impacter derrière les fonctions TMW permettant le traitement de ce paquet reçu.

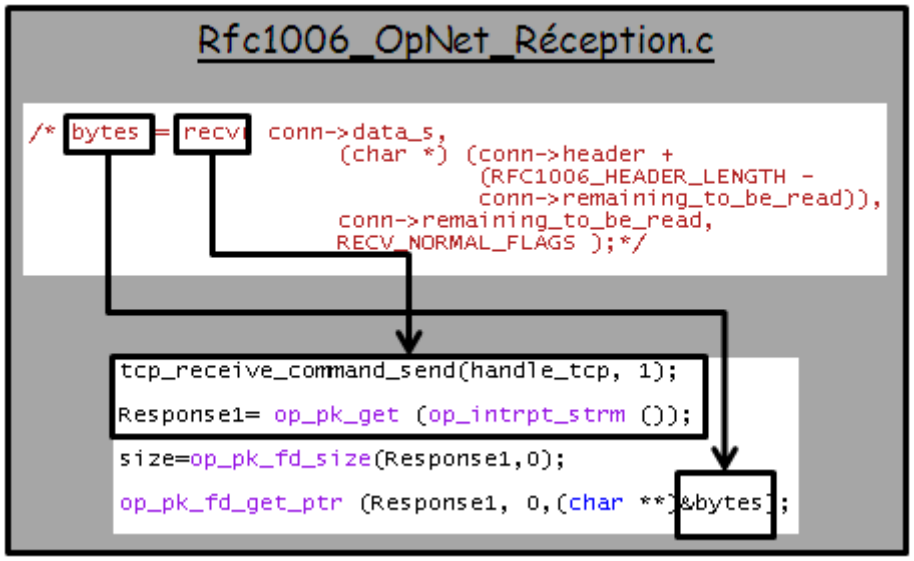


Figure 8.4 Adaptation de la réception des paquets MMS avec le stack TCP/IP d'OpNet

A l'issue de cette adaptation, une régénération de la librairie de fonction 1006.lib a été effectuée en tenant en compte les modifications du fichier source rfc1006.c. Cette librairie a permis aux messages MMS-IEC61850 construits pas les sources protocolaires de TMW d'être mappées sur le profil TCP/IP virtuel des modèles de simulation OpNet.

8.4 Résultat d'implémentation du profil MMS-TCP/IP dans la plateforme de Co-Simulation

Plusieurs fonctionnalités et services de communication existent dans le profil MMS-IEC 61850. Une fonctionnalité essentielle choisie pour cette première implémentation est celle permettant la configuration d'une application serveur IEC 61850 en fonction du fichier CID.

Le premier service de communication implémenté dans notre étude est le service **'Auto Discovery'** {chapitre 3, paragraphe 3.2.3}. Ce service est fourni par la norme IEC 61850 afin de permettre aux clients IEC 61850 de récupérer la base de données, décrite en format objet, contenue dans les serveurs [Santos, 2009], et ce dans le but de faciliter le développement des systèmes de supervision. Ce service possède une autre fonction utilisée par les outils de tests, afin de vérifier la conformité du contenu des applications IEC 61850 par rapport à leurs fichiers de description ou de configuration SCL (i.e. CID ou dans le cas échéant le ICD).

Cette modélisation a nécessité l'implémentation, dans l'environnement de la plateforme de Co-Simulation, de la librairie **'1006_opnet.lib'** issue du fichier rfc1006.c modifié et deux autres librairies intitulées **'cscl_opnet.lib'** et '**mmsc_opnet.lib**' (figure 7.5).

La première librairie **'cscl_opnet.lib'** contient des fonctions permettant le traitement des fichiers de configuration IEC 61850 'CID ou ICD' pour ensuite construire la base de données objets de l'équipement (i.e. serveur IEC 61850) en fonction du contenu de ces fichiers {Annexe D} [Haffar et al, 2010b].

La deuxième librairie '**mmsc_opnet.lib**' vise à fournir des fonctions permettant le traitement des services de communication MMS-TCP/IP. Le service **'Auto Discovery'** constitue l'une des fonctions informatiques contenue dans cette librairie. D'autres fonctions, permettant la lecture/écriture des attributs de données, existent aussi dans cette librairie.

La dernière librairie **'1006_opnet.lib'** consiste, comme nous l'avons déjà expliqué, à interagir avec le stack TCP/IP pour transmettre les services de communication fournis par la deuxième librairie.

Une architecture Co-Simulée SRR a été développée dans le but de permettre la validation de cette première phase de modélisation du standard IEC 61850 (figure 7.5). Cette architecture est composée du modèle 'MUT' qu'on souhaite tester, connecté à l'outil IEDScout de TMW {chapitre 7, paragraphe 7.2}, par l'intermédiaire de module HITL de notre plateforme {chapitre 4, paragraphe 4.5}. Le modèle a été configuré par l'intermédiaire d'un vrai fichier 'ICD' qui a été construit par le logiciel de configuration IEC 61850 Test Harness {chapitre 7, paragraphe 7.2}.

Le fichier SCL implémenté dans le MUT constitue un fichier de description **'Simple_Server.icd'** ne contenant aucune configuration avancée de communication IEC 61850 (i.e. Reporting, Goose). Pour simplifier sa construction, une fonctionnalité de mesure a été implémentée dans ce fichier. Cette fonctionnalité est représentée par le **'LN Measurement',** elle contient quatre objets de mesure : mesure de la puissance représentée par **'W'**, mesure de la fréquence représentée par **'Hz'**, mesure de l'énergie représentée par **'TotW'** et mesure de l'intensité de courant représentée par **'A'**. Chaque objet de mesure contient cinq attributs de données représentant respectivement, la valeur numérique de l'objet **'Analog'**, la qualité de la mesure **'Q'**, le temps de rafraichissement **'Time Stamp'**, la valeur maximale mesurée **'Upper Value'** et la valeur minimale mesurée **'Lower Value'**.

La figure 7.5 montre la validation de la réponse au service Auto Dicovery pour un MUT. La réponse a bien été reçue par l'outil de test IEDScout et le contenu de cette réponse est bien identique au fichier de configuration implémenté dans le modèle MUT [Haffar et al, 2010a].

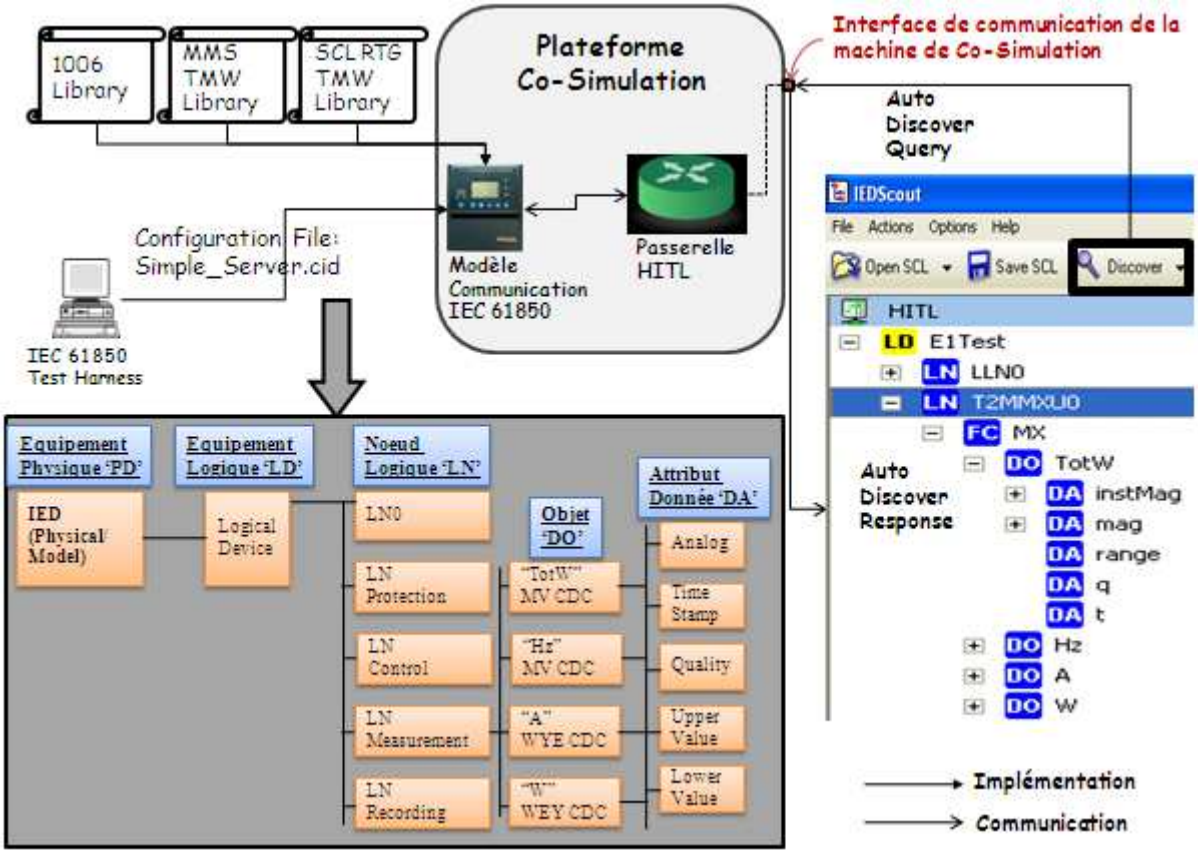


Figure 8.5 Architecture de validation de l'implémentation du service Auto-Discovery du profil MMS-TCP/IP

Plusieurs points peuvent être retirés de cette Co-Simulation :

En premier lieu, nous avons pu découvrir lors de cette modélisation qu'il faut effectuer des adaptations des fichiers sources ainsi que des fichiers de compilation livrés par le fournisseur TMW dans le but d'implémenter les sources codes du standard IEC 61850 dans les couches applicatives des modèles OpNet. Par conséquent, cette Co-Simulation nous a permis de vérifier la bonne adaptation des fichiers TMW par rapport à l'environnement de Co-Simulation OpNet.

Le deuxième point important à retirer de cette Co-Simulation est qu'on a pu configurer les modèles de communication OpNet avec des vrais fichiers de configuration IEC 61850. Par conséquent, ceci nous permet également de vérifier que la phase de FAT {chapitre 3, paragraphe 3.5.4} pourra être bien établie conformément aux besoins exprimés. En effet, les équipements absents lors de cette phase peuvent être remplacés par leurs modèles qui seront désormais configurés par les vrais fichiers de configuration de ces équipements absents.

Le troisième point à déduire est qu'on a pu, par le biais du module HITL de notre plateforme de Co-Simulation, valider la conformité du premier service d'auto découverte IEC 61850 implémenté dans nos modèles. La fonctionnalité d'auto découverte constitue l'un des services du profil MMS-TCP/IP du standard IEC 61850 {chapitre 3, paragraphe 3.2.4}.

Cette modélisation n'est qu'un premier pas pour une implémentation totale du standard IEC 61850 dans la plateforme de Co-Simulation. Comme nous l'avons montré lors de l'implémentation de ce service, les autre services doivent bien être validés et ce dans le but de

vérifier la conformité de nos modèles de communication par rapport à la spécification du standard de communication.

Plus spécifiquement, les services de communication, appartenant au profil MMS-TCP/IP, qui permettent la lecture/écriture des objets IEC 61850 doivent faire l'objet des futurs travaux à effectuer pour assurer une implémentation complète de ce profil. En plus, les autres profils GOOSE, SV, Synchro expliqué dans {chapitre 3, paragraphe 3.2.4} doivent de même être implémentés et validés afin d'arriver à un modèle complet certifié conforme à la norme IEC 61850.

Ce n'est qu'une fois que l'implémentation du standard IEC 61850 sera complètement réalisée dans la plateforme, qu'on pourra passer vers l'étape d'évaluation de performance en phase de conception et de vérification des SDC IEC 61850 en phase de FAT et de SAT.

8.5 Conclusion

Ce chapitre a présenté la première phase de modélisation du standard IEC 61850 dans la plateforme de Co-Simulation. Cette phase a montré l'implémentation des services de communication appartenant au profil MMS-TCP/IP.

Il a été clairement montré que la modélisation du standard IEC 61850 a nécessité des adaptations au niveau des fichiers de compilation et des fichiers sources livrés par le fournisseur des sources protocolaires.

Des outils de configuration et de test des services de communication IEC 61850 implémentés dans les modèles ont été présentés dans ce chapitre. La connexion entre ces outils et la plateforme de Co-Simulation a été établie par le biais du module HITL de cette plateforme. Ceci a permis la validation de l'un des premiers services de communication IEC 61850, permettant la découverte automatique de contenu d'un équipement.

En plus de l'implémentation et du test des services de communication, une fonctionnalité essentielle de la norme IEC 61850 a été programmée dans les modèles de Co-Simulation. Cette fonctionnalité permet la configuration des modèles, par les fichiers SCL, conformément au dictionnaire d'objets imposé par la norme IEC 61850.

La validation de cette fonctionnalité est liée, en quelque sorte, avec le premier service de communication implémenté. On a montré que la réponse d'un serveur suite à la sollicitation d'une requête d'auto-découverte contient l'ensemble des objets configurés dans ce serveur. Ainsi, le traitement de cette réponse par les outils de test a permis, d'une part, la validation du service de communication implémenté dans le modèle de Co-Simulation et d'autre part, la validation du contenu du modèle par rapport au fichier SCL implémenté dans ce modèle.

La suite de l'implémentation des autres services du profil MMS-TCP/IP ainsi que les autres profils de communication du standard Goose-Ethernet, SV-Ethernet feront l'objet des perspectives de travail de cette recherche, mais le cadre global de modélisation est maintenant disponible…

Chapitre 9

Conclusion générale et perspectives

9.1 Conclusion générale

Le nouveau standard de communication IEC 61850 a commencé à faire ses premières apparitions dans les différentes entreprises mondialement reconnues dans le domaine de la construction des appareillages électriques. Cet élan vient du fait que ce standard offre aux architectures de communication des installations de distribution électriques des points avantageux demandés par les clients finaux ; ne citons que les deux les plus principaux: l'interopérabilité et les échanges des messages de communication sécuritaires au travers de réseau de communication.

L'interopérabilité autorise la construction d'architectures de communication hybrides composées d'unités multi-constructeur ou l'extension d'architectures mono-constructeur existantes avec des composants venant d'autres constructeurs.

Le transfert des messages de communication sécuritaires est pourvu par la norme afin d'assurer un moyen efficace d'échange des informations de contrôle pour les applications d'automatismes distribués entre plusieurs unités de protection. Ce moyen d'échange est un point fortement demandé par les clients des installations électriques pour passer d'installations filaires à des installations en réseau.

Du fait de l'importance des applications de protections distribuées et de leur influence directe sur les attributs de la sûreté de fonctionnement des systèmes de distribution, la fiabilité des flux de données échangés dans ces nouveaux systèmes de communication doit être vérifiée tout au long des phases de développement d'un nouveau projet IEC61850. Plus spécifiquement, dès lors de la phase de conception de ces systèmes, une étude estimative de la performance des échanges de communication sécuritaire doit être effectuée pour valider le choix de l'architecture de communication et de ces flux de données. Cette étude doit être accomplie dans les pires cas pendant lesquels plusieurs évènements électriques apparaissent au même instant et contribuent à une avalanche des signaux sécuritaires au niveau du réseau de communication IEC 61850.

Cette performance doit être poursuivie en phase de tests du système de communication. Les tests se déroulent en deux phases afin de réduire au maximum le temps de la mise en service du système final. Les premiers tests, « **F**actory **A**cceptance **T**ests » **(FAT)**, sont réalisés après la finalisation du développement du système, généralement à l'emplacement du constructeur. La seconde phase, « **S**ite **A**cceptance **T**ests » **(SAT)**, consiste à les reproduire sur site juste avant la mise en exploitation du système. La réalisation des tests d'une telle architecture est en général irréalisable en phase de FAT du fait que chaque constructeur est responsable du développement d'une partie de l'architecture. Le problème de l'évaluation complète en phase de conception reste ouvert, et un tel scénario impose d'attendre le déploiement de

l'architecture sur site pour pouvoir la tester. Cela peut poser des problèmes lors du déploiement, et de la mise en service.

C'est pour toutes ces raisons que nous avons présenté dans notre étude le développement d'une plateforme d'évaluation de performance basée sur l'approche de Co-Simulation qui permet le remplacement des équipements absents en phase de tests FAT par des modèles ayant un comportement conforme en termes de communication. Les tests de performance seront ainsi réalisés sur une architecture Co-Simulée mixte autorisant les échanges de communication entre des modèles de simulation et des équipements réels.

Ces échanges exigent impérativement la validation de la conformité des protocoles implémentés dans les couches applicatives des modèles par rapport à la spécification du standard de communication en question. Des Co-Simulations ont été réalisées sur un protocole de communication libre de droit et répandu en industrie afin de montrer que cet aspect de validation peut finalement être réalisé en utilisant les modules de la plateforme de Co-Simulation.

Pour donner des mesures fiables des performances des architectures Co-Simulées, des études comparatives entre plusieurs architectures hybrides ont été réalisées afin de montrer l'impact des modules de notre plateforme sur les performances temporelles des échanges de communication entre les équipements réels et les modèles de simulation. Même qu'ils soient de l'ordre de 10 ms, ces retards contribuent à la non viabilité des signaux sécuritaires contribuant ainsi au dysfonctionnement des applications de protection distribuées associées. Le résultat de ces études nous a permis d'identifier les limites qui doivent être prises en compte lors de l'utilisation de cette plateforme de Co-Simulation pour les phases de vérification des systèmes de communication IEC 61850.

Le dernier chapitre a fait état de l'implémentation d'un des premiers services du profil MMS-TCP/IP dans les modèles de simulation IEC 61850 OpNet. Par ailleurs, ces modèles sont développés de manière à ce qu'ils soient capables de se configurer par le biais des vrais fichiers de configuration IEC61850. La validation de la modélisation de ces deux fonctionnalités essentielles du standard a été réalisée par le biais d'un équipement de tests externe à la plateforme et livré par le fournisseur TMW. Le module HITL de la plateforme de Co-Simulation a servi à ce stade comme une interface assurant un échange efficace des paquets entre l'équipement de test appartenant au monde réel et le modèle implémentant les fonctionnalités 61850 à tester.

9.2 Perspectives de ce travail de recherche

La modélisation du standard IEC 61850 effectuée à ce jour n'est qu'un premier pas pour une implémentation totale du standard IEC 61850 dans la plateforme de Co-Simulation. En théorie, l'adaptation des fichiers sources de TMW en fonction du stack TCP/IP de l'environnement de la Co-Simulation doit permettre le bon fonctionnement de tous les services du profil MMS tels que la lecture/écriture des objets et les échanges des fichiers lourds (i.e. rapport d'objets). Toutefois, ces fonctionnalités n'ont pas encore été testées. La première tâche en perspective, sera donc de terminer la validation des services du profil MMS-TCP/IP.

La deuxième perspective concernera la génération d'un fichier de configuration CID à partir d'une installation de distribution électrique réelle en utilisant l'éditeur de configuration standard SCL. En effet, les premiers tests utilisaient les fichiers de configuration ICD (fournis

par le constructeur). Ces fichiers ne contiennent que les fonctionnalités de l'équipement alors que le fichier CID contient également les services de communication associés à l'équipement.

Le troisième axe revient à assurer l'implémentation des profils Goose-Ethernet et Sampled Value-Ethernet dans les modèles de la plateforme. Cette tâche comprend :

- La modification de la passerelle HITL afin d'envoyer les trames directement de la couche application vers la couche Ethernet.
- Adaptation des fichiers sources TMW permettant l'envoi des requêtes événementielles afin de les rendre compatibles avec OpNet.
- Validation de la modélisation des Goose en connectant un serveur simulé à un équipement réel.
- Validation de l'ensemble des services (MMS et Goose) sur une petite architecture comprenant un client, deux serveurs simulés et un serveur réel.
- Validation de l'ensemble des services sur une architecture de distribution électrique multi constructeurs (type CERN).

Une fois tous les services des différents profils de communication IEC 61850 bien implémentés et validés, le travail pourra se poursuivre par l'évaluation des performances des architectures Smart Grid IEC61850 en vue de la validation de la plateforme en phase de conception. L'objectif est d'analyser la validité des modèles développés en termes de temps de réponse. Les résultats obtenus permettront de savoir si les modèles sont utilisables en phase de conception d'une architecture de distribution électrique. Pour cela, les temps de réponse entre une architecture 61850 complètement simulée et une architecture réelle devront être comparés.

Cette plateforme devra ensuite être validée en phase de vérification et de test (FAT) au sens des automatismes distribués. L'objectif étant de s'assurer que ces automatismes fonctionnent en gardant les mêmes performances en remplaçant un serveur réel par un serveur simulé.

Finalement, après avoir effectué toutes ces phases de validation cette recherche doit déboucher vers la phase d'utilisation de la plateforme pour valider le choix des architectures et des flux de communication en phase de conception en faisant des études probabilistes et statistiques (en produisant des phénomènes d'avalanche des signaux sécuritaires Goose).

Annexes

Annexe A
Librairie HITL

Cette annexe représente les détails de l'implémentation de la librairie. Le premier paragraphe montre le bloc de conversion HITL utilisé par les différentes fonctions de translation. Le deuxième paragraphe montre le schéma d'implémentation des fonctions de translation de la librairie HITL dans les attributs de la passerelle HITL.

En final le dernier paragraphe montre le détail de fonctions informatique OpNet utilisées dans les différentes versions des librairies.

Bloc de conversion HITL

Au lancement de chaque procédure de translation, un bloc de contrôle de conversion intitulé **'HITL Bloc de Conversion'** sera invoqué par toutes les fonctions de translation.

Ce bloc est formé d'une structure de donnée défini en langage C et intitulé **'sitlT_SCDB'**. Cette structure contient les variables nécessaires pour effectuer la translation des paquets dans les deux sens.

La figure A.1 montre les différents constituants de ce bloc :

```
typedef struct Sitl_Conversion_Descriptor_Block
        {
        SitlT_Translation_Direction  direction;
        Packet*                      sim_pk_ptr;
        unsigned char*               real_pk_ptr;
        unsigned int                 real_pk_size;
        unsigned int                 real_pk_offset;
        unsigned int                 max_conv_level;
        unsigned int                 cur_conv_level;
        unsigned int                 limit_passthru_to_ip;
        unsigned int                 drop_unsup_packets;
        unsigned char*               user_data_ptr;
        void*                        current_tfe_ptr;
        int                          passthrough;
        int                          link_layer_type;
        } SitlT_SCDB;
```

Figure A.0.1 Contenu du bloc de conversion

Parmi ces constituants, il existe ceux qui doivent être obligatoirement manipulés par les fonctions de translation et d'autres qui sont optionnels. Les champs obligatoires sont :

- Le champ **'direction'** indique aux fonctions de translation le sens de conversion
- Le champ **'sim_pk_ptr'** représente le pointeur vers la mémoire contenant le paquet simulé. Ce champ est considéré comme un paramètre d'entrée pour les fonctions de

translation direction Sim → Réel ou un paramètre de sortie pour les fonctions de translation direction Réel → Sim.

- Le champ **'real_pk_ptr'** représente le pointeur sur la zone mémoire contenant le paquet réel. Ce pointeur indique l'emplacement du premier octet du paquet réel. Il est considéré comme un point d'entrée pour les fonctions de translation Réel → Sim et un point de sortie à remplir par la fonction de translation SIM → Réel.

- Le champ **'real_pk_size'** indique la taille en octets du paquet réel reçu. Ce champ est nativement remplit par la passerelle HITL à chaque réception d'un nouveau paquet réel.

- Le champ **'real_pk_offset'** est un paramètre associé aux paquets réels. Il précise le décalage par rapport au premier emplacement à partir duquel les fonctions de translation peuvent se référencier. Ce champ doit être manipulé par des procédures d'incrémentation/décrémentation dans les fonctions de translation afin de mettre à jour l'emplacement d'écriture ou de lecture des nouvelles valeurs dans le paquet réel. Les fonctions de translation standards prennent en charge cette procédure après chaque lecture/écriture d'un nouveau champ. Toutefois, dans le cas d'une programmation d'une nouvelle fonction de translation pour les paquets applicatifs, cette manipulation doit être assurée par le programmeur afin de localiser l'emplacement des champs à lire ou à écrire.

Les champs optionnels associés à cette structure sont des paramètres utilisés pour assurer des fonctionnalités avancés de translation. Par exemple, le paramètre **'drop_unsup_packets'** indique à la passerelle l'autorisation du rejet des paquets non supportés par cette dernière.

Librairie de fonction et schéma d'implémentation

Pour assurer une procédure de translation complète des paquets, une librairie de fonction utilisateur doit être programmée pour assurer la conversion des paquets applicatifs. Cette programmation est effectué en langage C en utilisant l'éditeur **'External Source (C Code)'** du logiciel OpNet Modeler et compilé pour générer des fonctions exportés dynamiquement (DLL Export Functions).

Un schéma d'implémentation, montré sur la figure A.2, doit être bien respecté pour l'intégration de cette librairie ainsi que ses différentes fonctions dans l'environnement de la plateforme de Co-Simulation.

Une des premières implémentations consiste à implémenter le fichier C de cette librairie dans l'emplacement des librairies du logiciel de simulation OpNet Modeler. Ceci étant pour permettre le chargement des fonctions de cette librairie au démarrage d'un projet de simulation.

Les fonctions de translation doivent être implémentées dans les attributs de la passerelle HITL correspondante à la translation de la couche applicative.

La constitution de la librairie peut être séparée en deux blocs, le premier contient des procédures et des fonctions statiques tandis que le second contient des fonctions dynamiques.

Les fonctions statiques seront utilisées dans une fonction d'initialisation existante dans le bloc de fonction dynamique et seront chargées en mémoire lors de l'appel de cette fonction.

Les fonctions dynamiques possèdent le préfixe **'DLLEXPORT'**. Ces fonctions vont servir au remplissage des attributs de la passerelle HITL. Ces attributs sont considérés comme des points d'entrées à la passerelle permettant la translation des données relatives à la couche applicative. L'exécution de ces points d'entées se produit juste lors de la phase de la translation des données applicatives, d'où l'aspect dynamique de ces fonctions.

Les deux premiers attributs (i.e. fonctions du bloc dynamique) représentent les fonctions à utiliser pour effectuer la translation du stack se trouvant au dessous de la couche applicative. Tandis que le dernier attribut contient la fonction d'initialisation et a pour but d'effectuer la translation des données liées à la couche applicative.

La figure A.2 illustre le schéma d'implémentation des différents blocs de fonctions existants dans la librairie HITL.

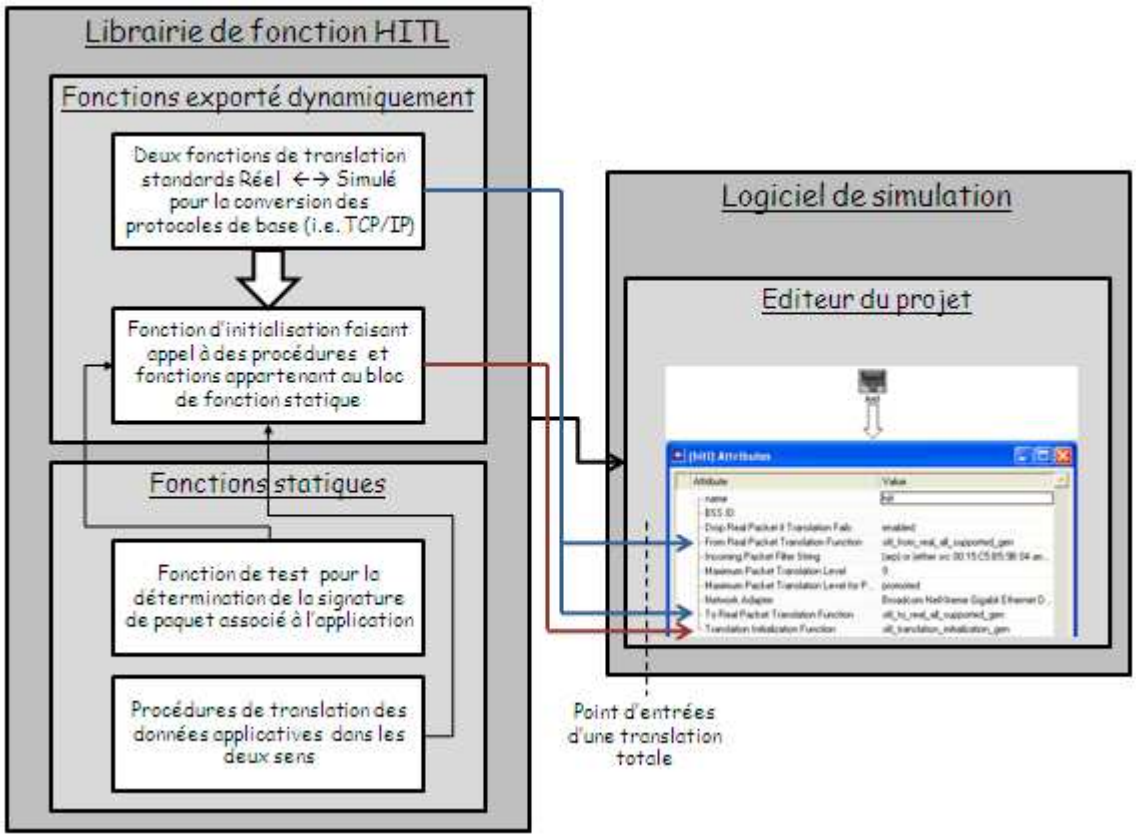


Figure A.0.2 Schéma d'implémentation d'une translation totale

Programmation de la première version de la librairie HITL

Comme toutes les autres versions, la première version est divisée en deux procédures. La première procédure consiste à la translation des paquets dans le sens simulation réel tandis que la deuxième est dans le sens inverse.

Ces deux procédures sont implémentées dans la fonction d'initialisation de la zone de fonction dynamique. Ainsi la fonction d'initialisation sera formée des deux fonctions de translation des paquets applicatifs permettant d'assurer la conversion dans les deux sens d'échange.

Procédure Simulation réelle

La translation simulé réelle, est basée sur les fonctions **'op_pk_nfd_get_pkt'**, **'op_pk_total_size_get'** et **'op_pk_fd_get_ptr'** se trouvant dans la bibliothèque des fonctions OpNet Modeler (figure A.3).

Figure A.3 Procédure de translation SR de la librairie HITL

La première fonction consiste en premier temps à extraire la partie relative à la couche applicative du paquet de simulation reçu par la passerelle.

La deuxième fonction permet la détermination de la taille du paquet non formaté reçu.

En final la troisième fonction, vise à copier les valeurs des champs de ce paquet dans une zone mémoire spécifique à la simulation. Cette zone est allouée à partir d'une adresse définie en tant qu'un paramètre de type 'void**' de cette fonction.

Après cette étape une boucle informatique de type 'for' est programmée pour transcrire la zone mémoire simulée déjà construite dans l'emplacement mémoire propre aux paquets réel du bloc de contrôle de translation **'real_pk_ptr'**.

En plus, une incrémentation de l'offset **'real_pk_offset'** du bloc de contrôle est assurée dans cette boucle informatique.

Procédure réelle Simulation (mettre le schéma de la librairie OpNet)

Etant donné qu'à la réception des paquets réels, les deux premiers points d'entrées assurent la translation des couches basses reconnues par la passerelle, le champ **'real_pk_offset'** sera ainsi mise à jour après chaque conversion. La zone mémoire à manipuler constituant le paquet associé à la couche applicative sera calculée à partir de l'adresse (**'real_pk_ptr'** + **'real_pk_offset')** et de taille égale **('real_pk_size' - 'real_pk_offset'). 'real_pk_ptr', 'real_pk_offset' et 'real_pk_size'** sont des champs accessibles dans le bloc de contrôle de translation HIT.

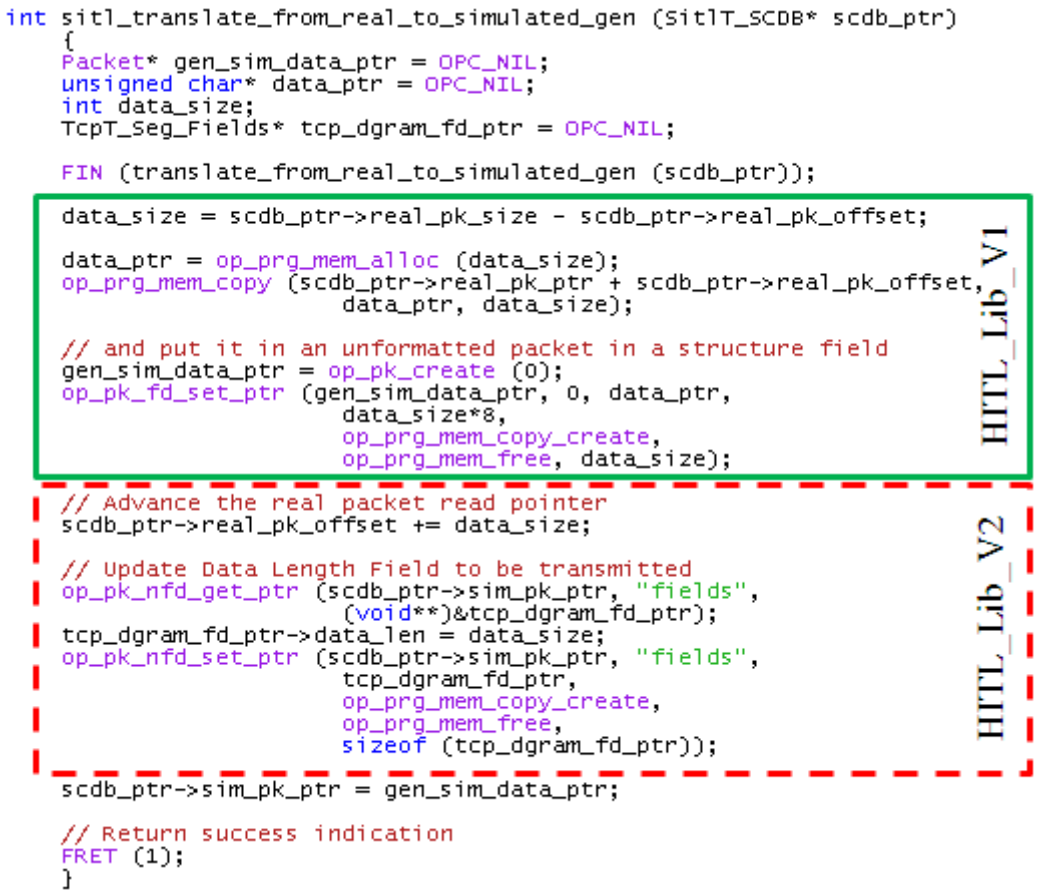


Figure A.4 Procédure de translation RS de la librairie HITL

La fonction OpNet **'op_prg_mem_copy'** est utilisée pour copier cette zone mémoire associée à la couche applicative dans une zone mémoire propre à la simulation (figure A.4).

La deuxième fonction permet la création du paquet de simulation par l'intermédiaire de la fonction OpNet **'op_pk_create'** et le remplissage de ce paquet par la zone mémoire déjà construite. Ce remplissage est effectué en utilisant la fonction OpNet **'op_pk_fd_set_ptr'**.

Programmation de la deuxième version de la librairie HITL

La deuxième version de la librairie a consisté à la mise à jour d'un champ TCP non convertit lors de l'utilisation des fonctions de translation des couches basses. Ce champ est celui qui indique la taille de données applicatives.

Pour cela, la méthodologie a consisté en premier temps d'extraire une structure intitulée 'fields' du segment TCP OpNet. Cette structure contient plusieurs champs du segment TCP dont le champ **'Data Length'** fait partie. Ce champ est celui qui n'est pas mis à jour avec les fonctions de translation pourvues par OpNet.

L'extraction de cette structure est accomplit par l'intermédiaire de la fonction **'op_pk_nfd_get_ptr'** (figure A.4).

La modification du champ **'data length'** est accomplit par un accès aux données associées à la structure 'fields'. La mise est à jour de ce champ est effectué en fonction de la taille des données associés à l'application. En final, la structure modifiée contenant la nouvelle valeur,

mise à jour, de ce champ **'data length'** sera ensuite insérée dans le segment TCP en utilisant la fonction **'op_pk_nfd_set_ptr'** (figure A.4).

Programmation de la troisième version de la librairie HITL

La troisième version de la librairie à viser un changement au niveau du type informatique du paquet de simulation envoyée vers la passerelle HITL.

L'appel aux deux fonctions de base **'op_pk_total_size_get'** et **'op_pk_fd_get_ptr'** par la première version de la librairie, a conduit à un échec de translation du fait que ces fonctions prennent en paramètres un champ de type **'packet *'**, et le paquet envoyé par la couche applicative du modèle de simulation est de type **'segment*'** (figure A.3).

Pour cela, la première étape a consisté à extraire la liste des paquets parentaux du segment envoyé.

Cette procédure est faite par l'utilisation de la fonction '**op_sar_seg_parent_packets_access ()'** prenant en paramètre un pointeur sur un segment de paquets et donnant en retour la liste des parents du segment envoyé (figure A.3).

En langage informatique, cette liste de paquet sera associé à un type intitulé par **'list *'**. En prenant comme supposition qu'un segment est issue d'un parent unitaire, la deuxième fonction de cette procédure a constitué d'extraire l'élément situé au premier emplacement de la liste déjà constituée.

Pour cela la fonction **'op_prg_list_access'** a été utilisée dans le but de détacher le premier élément de la liste des paquets parentaux (figure A.3).

Cet élément aura ainsi comme type **'packet *'** et correspond au paquet envoyé par la couche applicative du nœud de simulation.

Les fonctions de translation utilisée au niveau de la version 1 de la librairie HITL peuvent avoir effet sur le paquet déjà formé.

Annexe B
Modélisation du Protocole ModBus dans OpNet

Protocol de communication ModBus

Le protocole ModBus/TCP est un protocole applicatif existant au niveau des couches 5, 6 et 7 du modèle OSI (i.e. couche 5 du modèle TCP/IP). Ce protocole travaille sur le port 502 de la couche TCP, il est implémenté dans les serveurs et les clients pour permettre un échange de requêtes et des réponses (Figure B.1).

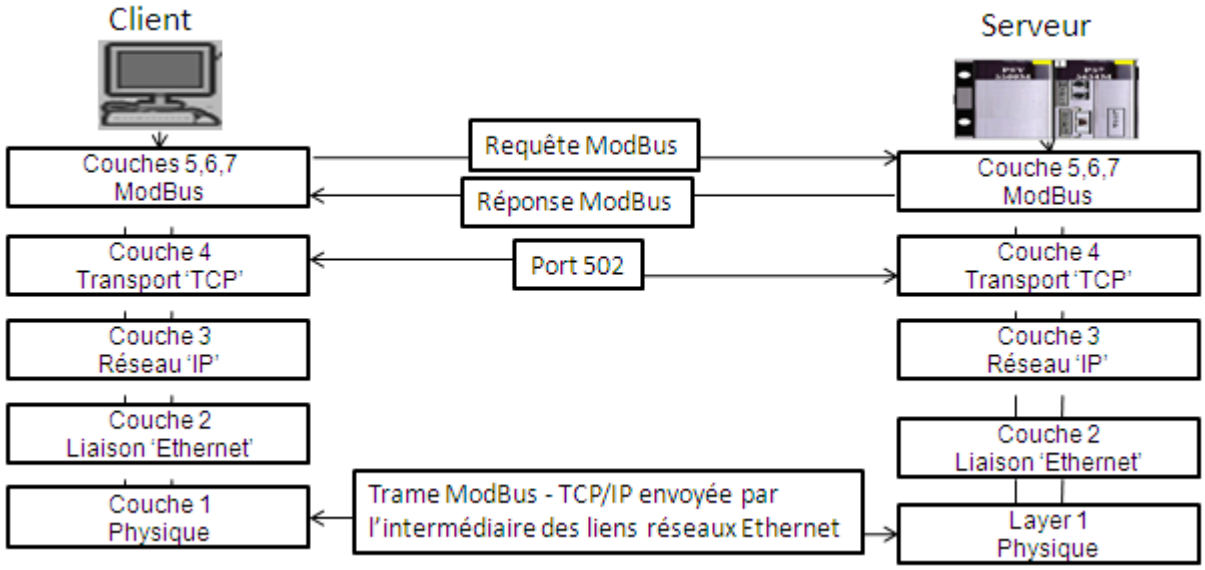


Figure B.0.1 Protocole ModBus dans les couches de communication des clients et des serveurs

Le format global d'un paquet ModBus est montré dans la figure B.2. L'entête de ce paquet est formé des quatre champs **'Transaction Identifier'**, **'Protocol Identifier'**, **'Length'** et **'Unit ID'**. 'Transaction Identifier' est un champ qui permet d'identifier la correspondance entre les requêtes et les réponses. 'Protocol Identifier' est un champ qui a une valeur fixe correspondante à identifier le protocole ModBus/TCP. La valeur du champ 'Length' est calculée en fonction de la taille en Octets des champs successeurs. En final le champ 'Unit ID' est utilisé dans le cas ou le client interroge un concentrateur de plusieurs équipements ModBus.

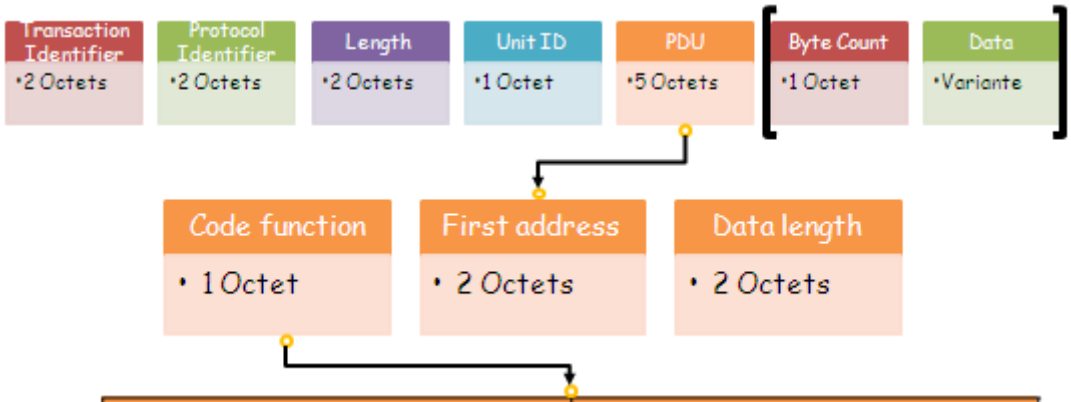
Le PDU est un sous paquet contenu dans le paquet global, qui indique les caractéristiques de la fonctionnalité de communication ModBus. Plusieurs fonctionnalités, montrées dans la figure B.2, sont pourvues par le protocole ModBus. Ces fonctionnalités sont identifiées par l'intermédiaire du champ **'Code Function'** du paquet PDU. Les fonctionnalités montrées en gras de cette figure sont celles qui sont implémentées dans nos modèles clients et serveurs. En effet, les fonctionnalités 4, 5 et 6 sont ignorées du fait qu'elles peuvent être remplacées par les

fonctionnalités 15 et 16. La fonctionnalité 176 est de même ignorée du fait que les simulations faites dans notre étude n'implémentent aucun concentrateur de données.

Les champs **'First Adress'** et **'Data Number'** indiquent l'emplacement et le nombre des données à manipuler.

Les champs **'Byte Count'** et **'Data'** appartenant aux deux parenthèses de la figure B.2 peuvent apparaître ou disparaitre en fonction de la fonctionnalité de communication ModBus (i.e. valeur du champ Code Function).

Si la fonctionnalité de communication ModBus correspond à une requête d'écriture ou à une réponse pour une requête de lecture, ces deux champs apparaitront dans le paquet envoyé. Dans le cas contraire, ces deux champs seront omis de ce paquet. Le champ 'Data' contient les données à écrire dans le serveur ou lues par le client. Le 'Byte Count' est un champ qui contient le nombre en octets de ces données.



Code Function	Fonctionnalité de communication
1	**Lecture n Bits**
3	**Lecture n Mots**
4	*Lecture registres entrées Mots*
5	*Ecriture 1 Bits*
6	*Ecriture 1 Mot*
15	**Ecriture n Bits**
16	**Ecriture n Mots**
17	*Lecture ID esclave*

Figure B.0.2 Format Global d'un paquet ModBus

Les formats de paquets OpNet

Deux formats de paquet existent dans le logiciel de simulation OpNet : Paquet formaté et paquet non formaté.

Les paquets formatés, pourvues par le logiciel OpNet, permettent de faciliter la construction des protocoles de communication et ce, en facilitant l'accès aux champs constitutifs de ces

paquets. En effet, dans le cas d'un paquet non formaté, l'accès se fait par l'intermédiaire des pointeurs mémoire tandis qu'en formaté ces champs auront un nom bien défini, attribué dans l'éditeur de construction des paquets OpNet.

La figure B.3 illustre la différence entre les deux modes d'accès. Un exemple est donné pour un accès à l'écriture du champ 'Transaction Identifier' du protocole ModBus. Dans le cas formaté, l'accès au champ TID a été fait par l'appel au nom défini dans le format du paquet formaté PKT_WRITE. Tandis que dans le deuxième cas, on a été mené à allouer une mémoire et à faire l'accès aux champs par rapport à leur emplacement dans la mémoire allouée.

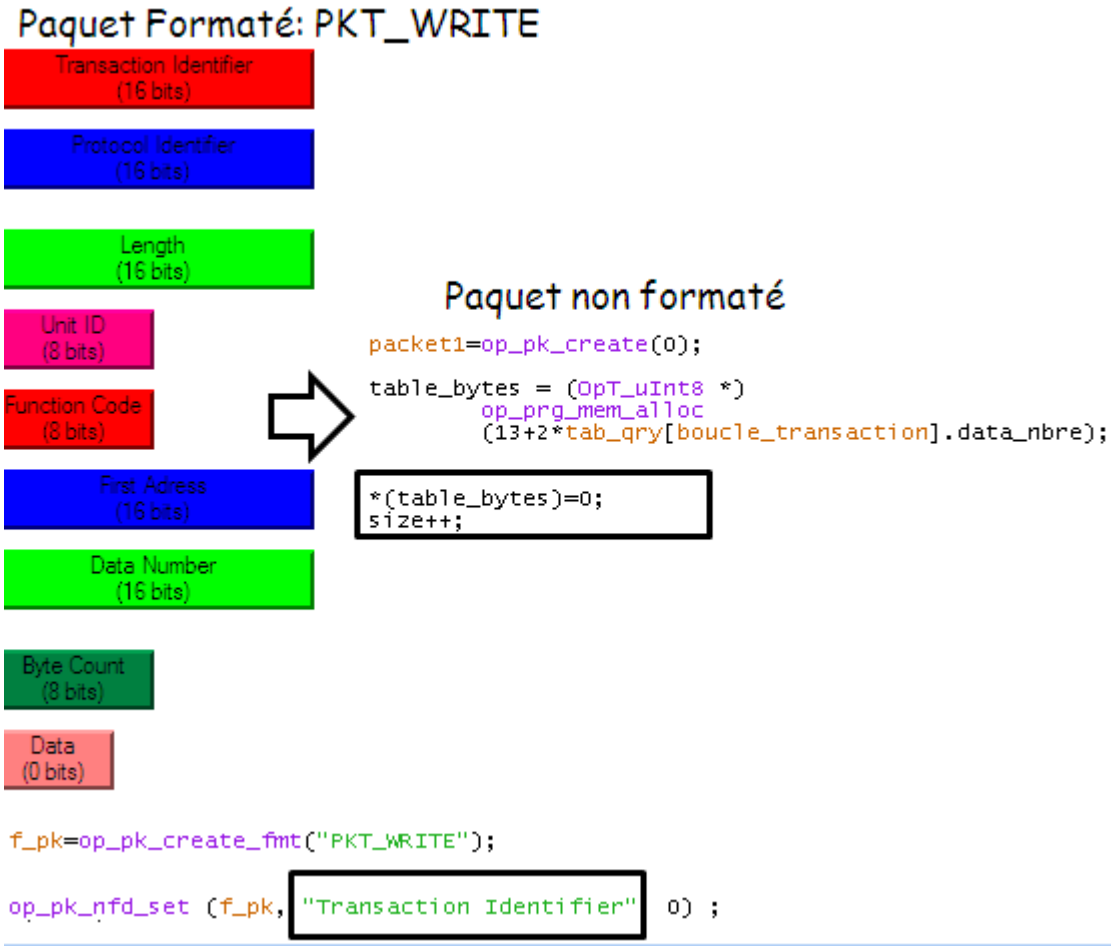


Figure B.0.3 Accès à un paquet formaté et non formaté

Traitement et construction des paquets ModBus/TCP avec le logiciel OpNet Modeler

Ce paragraphe illustre les différentes fonctions OpNet qu'on a utilisé dans le développement des Process client et serveur ModBus pour l'identification et la création des paquets ModBus.

Process applicatif du modèle serveur

Le développement du modèle serveur a consisté au traitement des requêtes client et la construction des réponses correspondantes à ces requêtes.

La figure B.4 détaille les fonctions OpNet utilisé pour l'identification des requêtes ModBus/TCP envoyé par un client.

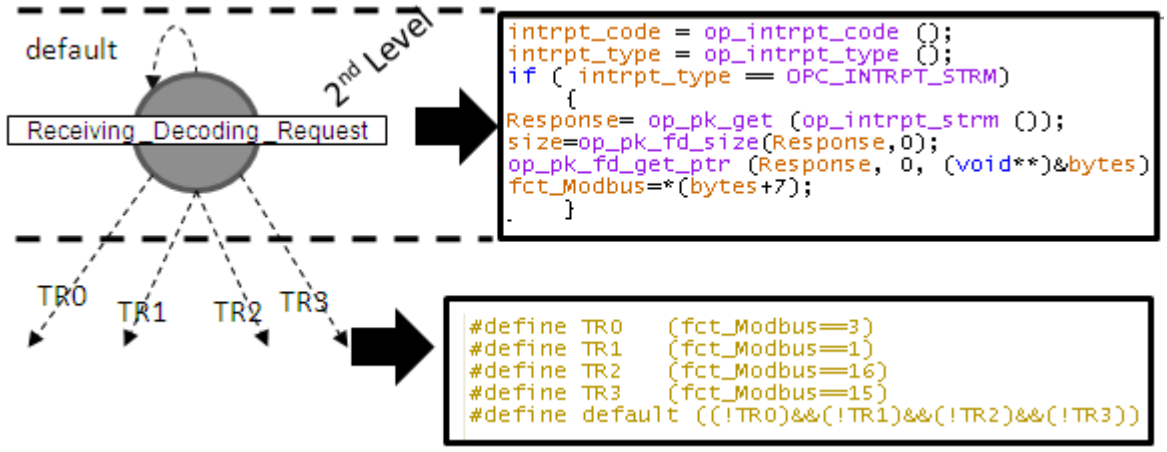


Figure B.0.4 Détails de programmation de l'étape d'identification des requêtes

On peut bien constater que la première chose effectuée dans l'étape du niveau 2 du modèle serveur, a consisté à la détection de la nature de l'interruption débloquant le Process par l'intermédiaire des fonctions **op_intrpt_code ()** et **op_intrpt_type ()**. Si le résultat indique une interruption provenant de la couche TCP **(OPC_INTRPT_STRM)**, le Process procèdera à la détermination de la nature du paquet envoyé par la couche TCP.

Etant donné que le paquet est envoyé par un client réel, ce paquet sera considéré comme étant de type non formaté. La fonction **op_pk_get ()** permet l'extraction du paquet non formaté correspondant à l'interruption détectée. La fonction **op_pk_fd_get_ptr ()** permet l'extraction de la structure des données contenue dans le paquet non formaté.

Cette structure a été rangée dans la mémoire allouée sous le nom de **bytes**. Pour vérifier que le paquet envoyé est bien de type ModBus et déterminer la fonctionnalité correspondante à ce paquet, un accès au 7^{ième} octet de cette structure a été effectué.

Cet Octet correspond au champ 'Code Function' du ModBus.

La transition vers le niveau 3 constituant la construction des réponses ModBus a été programmée dans le cas ou le champ 'Code Function' aura l'une des valeurs 1, 3, 15 ou 16. Ces valeurs correspondent aux quatre fonctionnalités de lecture écriture implémentées dans le serveur.

Pour expliquer le contenu des fonctions OpNet utilisées dans la construction des réponses ModBus au niveau 3 du Process serveur, on a montré dans la figure B.5 l'exemple de la construction de la réponse lecture informations analogiques.

Cette construction peut être séparée en trois blocs essentiels. Le premier montré en rouge permet la récupération des champs 'First Adress' et 'Data Number' de la requête envoyée. Les valeurs de ces champs vont servir à la construction de la réponse.

Le paquet 'Packet 1' constitue la réponse à envoyée, il est créé, dans le premier bloc vert de la figure, en utilisant la fonction OpNet de création des paquets non formaté d'OpNet **'op_pk_create'**. L'utilisation de ce type dans notre étude revient du fait que la passerelle HITL est programmée de manière à convertir les paquets de type non formaté.

Dans le bloc noir pointillé, une allocation mémoire de la structure des données constituant la réponse est effectuée. La taille ainsi que les valeurs de chaque champ mémoire constituant cette structure est remplit en fonction de la requête envoyé par le client.

Le dernier bloc montré en vert dans la figure, consiste à implémenter la structure des données mémoire déjà construite, dans le paquet non formaté créé au début. Ceci a été accompli en utilisant la fonction OpNet **op_pk_fd_set_ptr**. A la fin de ce bloc, la fonction **tcp_data_send** de l'API de TCP a été utilisée pour envoyer la réponse au client destinataire. De même, la fonction **tcp_receive_command_send** a été utilisée dans le Process applicatif du serveur dans le but d'informer la couche TCP qu'il est capable de recevoir d'autres requêtes du client.

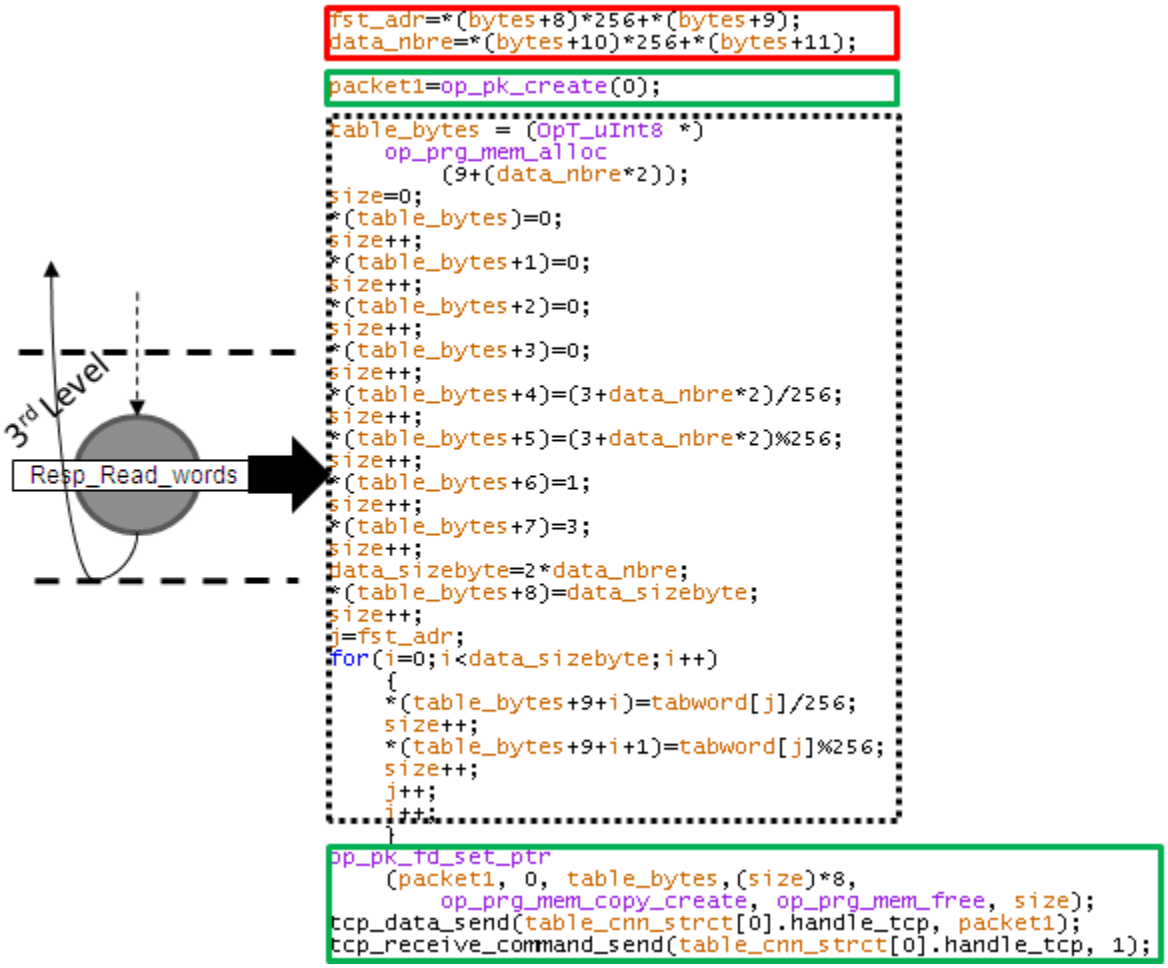


Figure B.0.5 Construction de la réponse lecture informations analogiques dans le modèle serveur

Process applicatif du modèle client

La première étape importante dans le développement du modèle client persiste au niveau 2 de son Process. Ce niveau contient le cœur du modèle qui permet l'interaction entre le Process applicatif et le module SITL.

La figure B.6 montre les différentes fonctions informatiques, issues de la librairie SITL, utilisées pour assurer cette interaction. Ces fonctions peuvent être classées en cinq blocs différents.

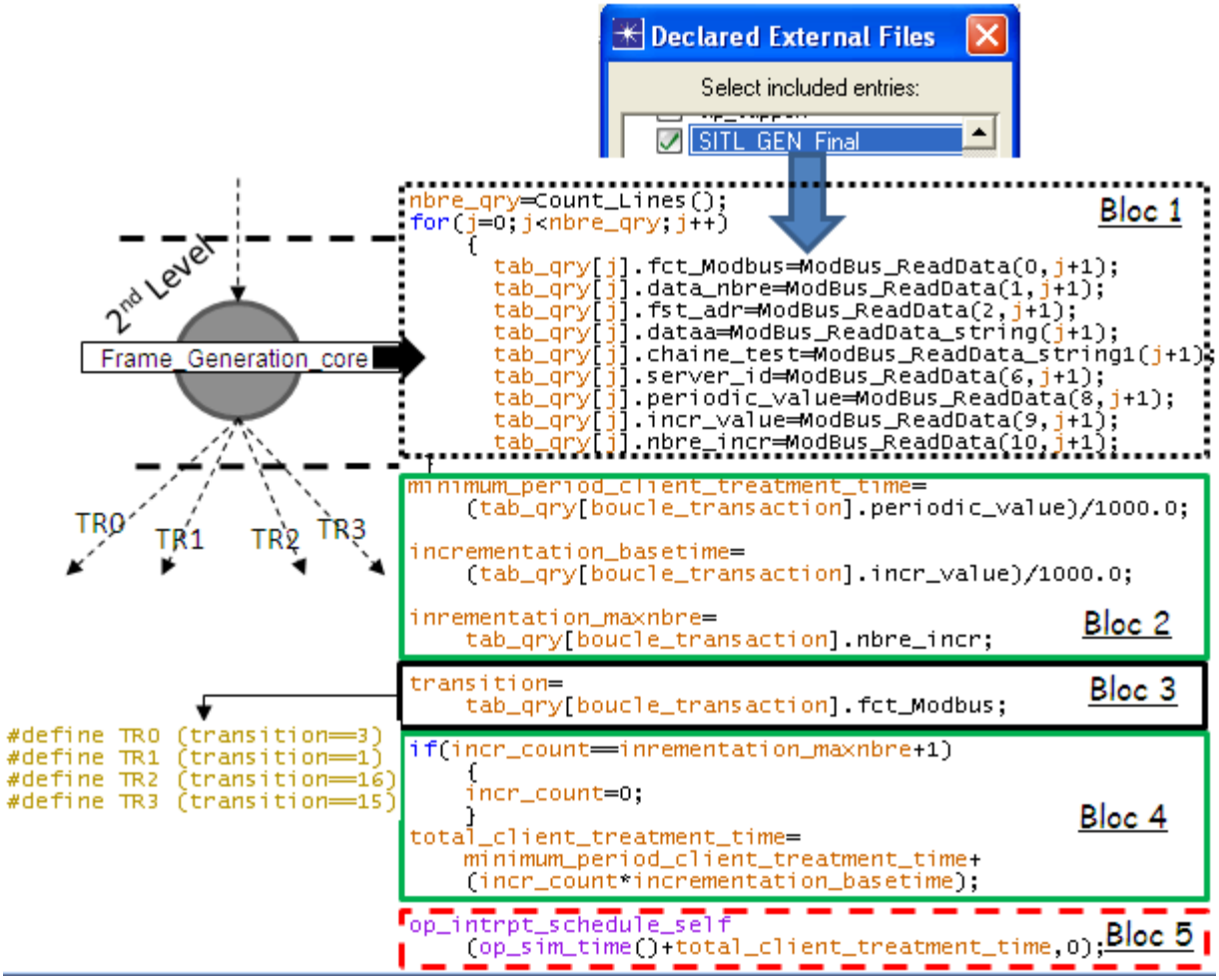


Figure B.0.6 Fonctions d'interaction entre le Process applicatif client et le module SITL

Le 'Bloc 1' consiste à établir une connexion vers la base de données SITL dans le but de récupérer toutes les informations liées à la génération des requêtes client. Ces informations seront placées dans un tableau de structure OpNet intitulée **'tab_qry'**.

Les deux blocs 'Bloc 2' et 'Bloc 4' consistent à calculer le temps de génération des requêtes en fonction des informations existantes dans le tableau **'tab_qry'**.

Le ‘Bloc 3’ permet de déterminer la nature de la requête demandée par l’application SITL. Ce bloc indique au Process la transition qu’il doit prendre lors de la réception d’une interruption de génération de requêtes. Ceci est réalisé en activant l’une des conditions, TR0 ou TR1 ou TR2 ou TR3, associées à chacune des transitions sortant du niveau 2. Comme le montre la figure B.6 ces conditions sont évaluées en fonction de la valeur de la variable **‘transition’** calculée dans le Bloc 3. Cette variable contient la valeur du champ ‘Code Function’ récupéré de la base de données SITL.

Le dernier bloc ‘Bloc 5’ permet la planification des interruptions Process correspondante en fonction des informations calculées dans les quatre premiers blocs (i.e. la nature des requêtes clients et leurs temps de génération).

A l’issue de la transition vers le niveau 3, le Process procède à la construction des requêtes correspondantes à la fonctionnalité ModBus demandée par l’utilisation via le SITL. La figure B.7 montre la construction de la requête ‘Lecture Informations analogiques’.

Ce qu’il faut noter est que la construction des requêtes client sont réalisées par l’intermédiaire du type non formaté et ce du fait que la passerelle HITL est programmée de manière à accepter juste les paquets non formaté.

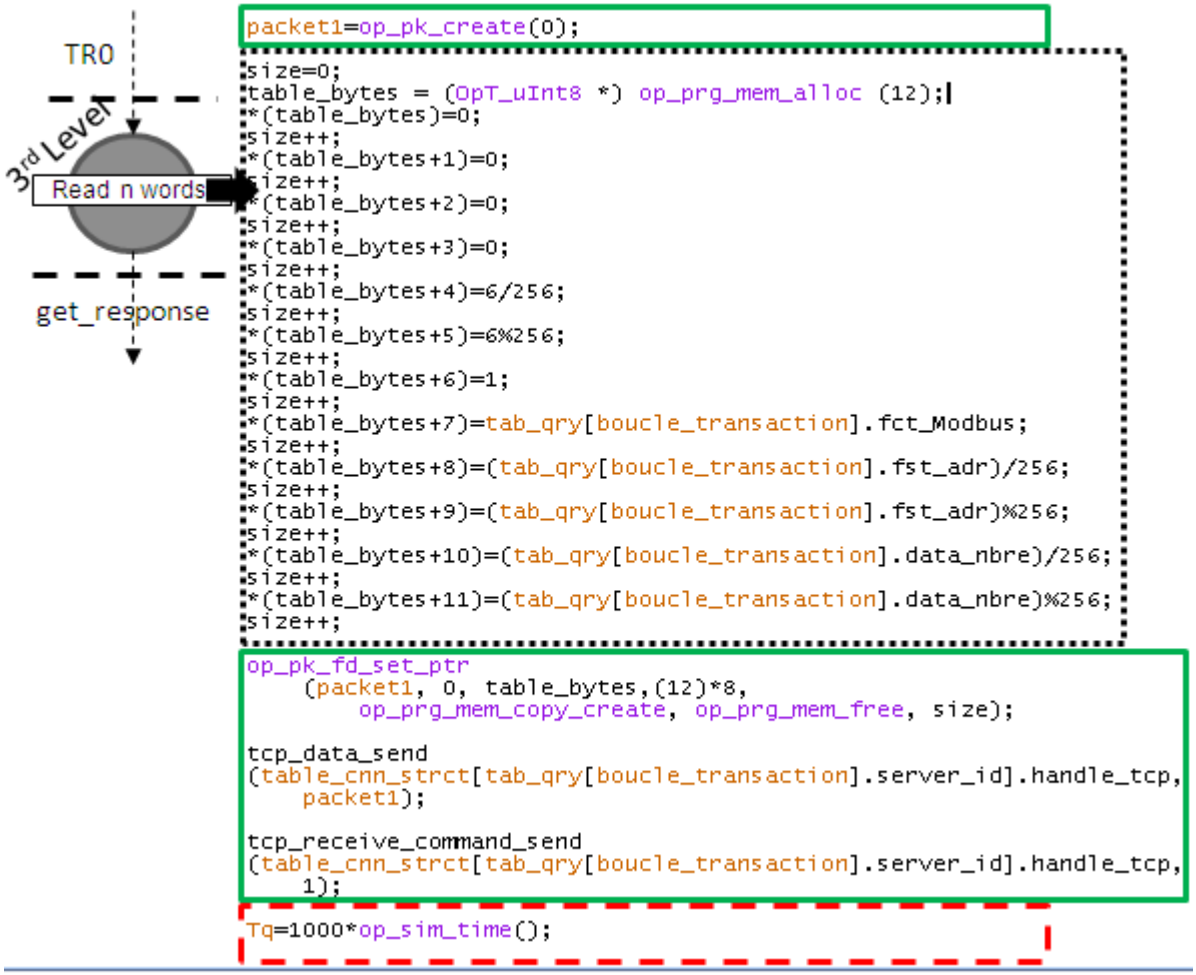


Figure B.0.7 Programmation de la requête lecture informations analogiques

Le premier bloc de cette figure consiste à la création du paquet non formaté. Le deuxième bloc permet l’allocation mémoire d’un tableau d’octet et son remplissage en fonction des

caractéristiques de la requête demandée. Le troisième bloc permet l'implémentation de la structure déjà formée dans le paquet non formaté créé dans le premier bloc. Il permet de même l'envoi du paquet construit vers la couche TCP du client en utilisant la fonction 'tcp_data_send'. A la fin de ce bloc, le Process envoie une indication à la couche de transport en lui informant qu'il est prêt à recevoir la réponse du serveur. Le dernier bloc consiste à enregistrer le temps d'envoi du paquet pour calculer le paramètre de performance 'délai de transaction' des échanges de communication.

Lorsque le client envoie sa requête, il bloque son Process en attendant la réception de la réponse du serveur. Ceci est informatiquement traduit par la détection d'une interruption venant de la couche TCP.

La figure B.8 montre la transition conditionnée par la variable 'get_response' correspondante à cette interruption.

A l'issue de la détection d'une telle interruption, le Process procèdera à l'extraction du paquet non formaté et de la structure contenue dans ce paquet. Cette étape est réalisée par l'intermédiaire des deux fonctions 'op_pk_get' et 'op_pk_fd_get_ptr' montrées dans le deuxième bloc de la figure.

Le troisième bloc permettra d'afficher le résultat de la réponse du serveur. Dans notre cas, les valeurs des informations analogiques demandées par le modèle client seront affichées sur la console OpNet en utilisant la fonction **'Printf'**.

Le dernier bloc permet d'incrémenter la boucle dans le but de prendre les autre requête contenues dans le tableau **'tab_qry'** du niveau 2.

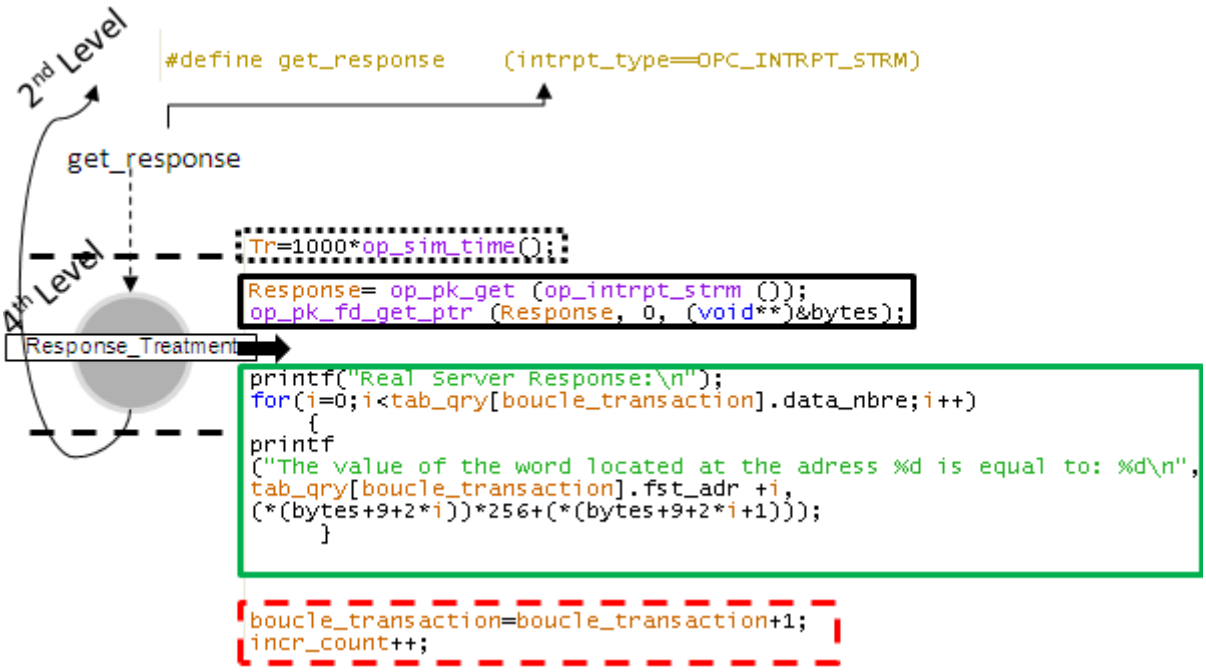


Figure B.0.8 Traitement des réponses des serveurs

Annexe C
Principe des MakeFile

Principe de fonctionnement

Les Makefile sont à la base des fichiers de compilation programmés avec une syntaxe bien particulière via des éditeurs textes. Ces fichiers possèdent comme extension **'.mak'**.

Des utilitaires type MAKE, NMAKE existent dans les systèmes d'exploitation pour autoriser l'exécution de ces fichiers et générer les sorties correspondants à ces derniers.

Plusieurs règles constituent un fichier de compilation Makefile. Chaque règle est formée d'une cible, des dépendances et des commandes.

Une cible est le fichier de sortie qui doit être généré après l'exécution du fichier de compilation.

Les dépendances sont les fichiers constituant la cible du fichier de compilation.

Les commandes contiennent des appels vers des compilateurs ou des éditeurs de lien afin de traiter toutes les dépendances et générer la cible correspondante à la règle du fichier de compilation.

Les commandes d'éditions de lien sont appelées après compilation des fichiers dans le but d'assurer un lien entre les différents fichiers compilé et construire les cibles correspondantes. Ces commandes doivent respecter la syntaxe imposée par le compilateur et l'éditeur de lien utilisés par l'environnement de développement de la cible.

A son tour, la syntaxe de programmation des fichiers de compilation dépend de l'utilitaire d'exécution, existant dans l'environnement de développement de la cible.

La première règle rencontrée est celle qui sera traitée par l'utilitaire de l'exécution des Makefile. La syntaxe d'écriture d'une règle consiste à séparer la cible et les dépendances par le symbole **':'**. Les commandes seront rédigées dans la ligne qui suit la ligne des cibles et des dépendances. D'autre part, les variable Makefile sont définis en utilisant le symbole **'='**. La figure C.1 montre les différents constituants d'une règle.

```
cible: dependance
 commandes
```

Figure C.0.1 Les trois composants constitutifs d'une règle Makefile

Les règles sont traitées par l'utilitaire d'exécution des Makefile (i.e. utilitaire Make de Linux ou utilitaire Nmake de Windows). Le principe de traitement est illustré sur la figure C.2.

Figure C.0.2 Traitement des règles Makefile

Si une dépendance constitue la cible d'une autre règle du Makefile, cette règle sera traitée par l'utilitaire Makefile.

Exemple d'un Makefile

Pour mieux comprendre le principe de fonctionnement des Makefile, un petit exemple de génération d'un exécutable intitulé **'hello.exe'** [Tutorial Makefile] est montré sur la figure C.3.

La première règle de ce Makefile est celle qui consiste à générer cet exécutable à travers la commande de compilation et d'édition de lien **'gcc'**. Les dépendances de cette règle sont les fichiers compilés ('hello.o' et 'main.o'). Ces fichiers sont les cibles d'autres règles qui seront ainsi traités en premier.

'hello.o' est un fichier objet généré par la commande de compilation 'gcc' à partir du fichier source 'hello.c'.

De même 'main.o' est l'autre fichier ojet généré après la compilation des fichiers 'main.c' et 'hello.h'. Si les deux commandes de compilation ne détectent aucune erreur dans les fichiers sources '.c', les fichiers objets seront générés.

Le lien entre ces deux fichiers objets compilé permet l'obtention de l'exécutable 'hello.exe' constituant la cible de la première règle de ce Makefile.

Makefile

```
hello: hello.o main.o
 gcc -o hello hello.o main.o

hello.o: hello.c
 gcc -o hello.o -c hello.c -W -Wall -ansi -pedantic

main.o: main.c hello.h
 gcc -o main.o -c main.c -W -Wall -ansi -pedantic
```

Figure C.0.3 Makefile de l'application exécutable hello.exe

Annexe D
Adaptation IEC 61850 TMW avec l'environnement de la plateforme de Co-Simulation

Afin d'utiliser les fonctions informatiques permettant la génération des services de communication, TMW fournit des fichiers de sources programmé en langage 'C'. Ces sources doivent être compilées par les utilisateurs afin de générer les librairies de fonctions informatiques permettant le développement des applications IEC 61850.

Des fichiers de compilation, de type Makefile, sont mis à disposition par le fournisseur TMW. L'exécution de ces fichiers est assurée par un utilitaire spécifique qui prend comme dépendances les fichiers sources 'C' et génère en sortie les cibles correspondantes à ces fichiers {Annexe C}. Les cibles TMW, appelées aussi les fichiers de sorties {chapitre 7, paragraphe 7.3}, sont soit des librairies de fonctions qui servent au développement des applications conforme IEC 61850 soit des applications exécutables pourvues par le fournisseur dans le but de donner des exemples des applications IEC61850.

On a expliqué dans {Annexe C} que les commandes ainsi que la syntaxe de programmation des fichiers de compilation dépend entièrement de l'environnement de développement de l'application IEC 61850. Par conséquent, plusieurs niveaux d'adaptation ont été requis pour rendre les fichiers de compilation compatibles avec l'environnement de développement de la plateforme de Co-Simulation.

Adaptation des fichiers de compilation système TMW

La première étape de l'adaptation a consisté à la configuration de l'environnement de développement de notre application IEC 61850 (i.e. environnement de développement de la plateforme de Co-Simulation). Chaque environnement possède ses propres caractéristiques dont les plus essentielles sont le type de compilateur, les éditeurs de liens et l'utilitaire d'exécution des fichiers de compilation Makefile (figure D.1).

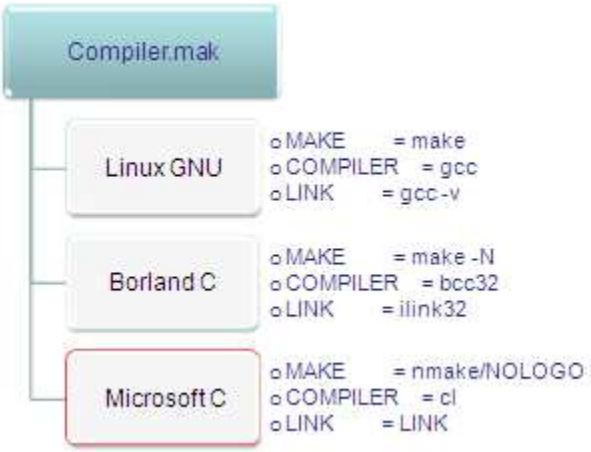


Figure D.0.1 Caractéristiques d'un environnement de développement

La configuration de l'environnement de développement a été effectuée en appliquant des adaptations au niveau des fichiers de compilation systèmes livrés par TMW. Ces derniers sont répertoriés en trois fichiers distincts 'Compiler.mak', 'Options.mak' et 'Master-Makefile.mak' (figure D.2). Les caractéristiques de l'environnement de développement font partie du fichier 'Compiler.mak'.

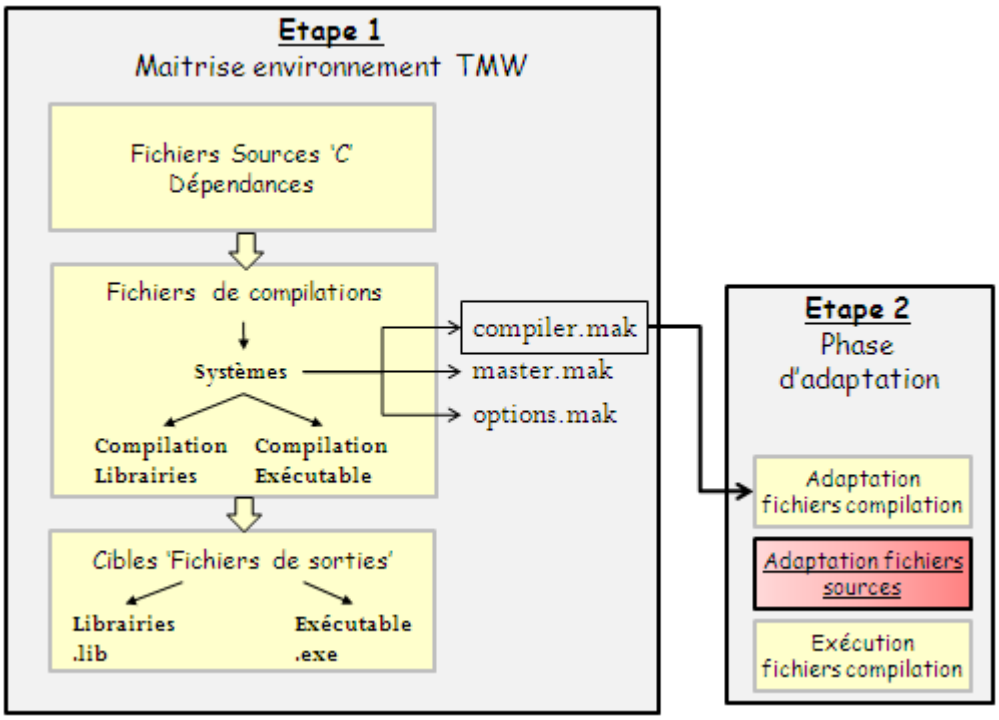


Figure D.0.2 Répartition des fichiers TMW et adaptation des fichiers systèmes

Etant donné que notre plateforme de Co-Simulation tourne sous Windows et utilise le compilateur et l'éditeur de lien du Microsoft Visual Studio, l'utilitaire d'exécution des Makefile sera celui utilisé par le Windows **'NMAKE.exe',** et le compilateur et l'éditeur du lien seront ceux utilisé par l'éditeur de programmation Visual Studio, **'cl.exe'** et **'link.exe'**.

Pa conséquent, la figure D.3 montre les différentes configurations que nous avons effectuées au niveau du fichier compiler.mak. On peut bien remarquer que ce fichier ne contient aucune programmation des règles Makefile. Toutefois, si les variables contenues dans ce Makefile doivent être bien définies du fait qu'elles seront utilisées dans tous les autres fichiers Makefile qui permettent en particulier la génération des sorties Lib et Exe (figure D.2).

La première variable 'MAKE' a été configurée de manière à faire appel à l'utilitaire '**nmake.exe'** (figure D.3). La variable 'Compiler' contient la valeur 'cl.exe' dans le but de faire appel au compilateur de la plateforme et finalement la variable 'link' fait appel à l'éditeur de la plateforme 'link.exe' avec les options associées à cet éditeur de lien.

D'autres variables existent dans ce fichier permettant de définir les extensions des sorties générées par les Makefile.

La variable OBJSUFFIX indique l'extension des fichiers objets généré après une commande de compilation d'un fichier source .C. Cette variable contient la valeur **obj**, indiquant l'extension de ces fichiers dans Windows. De même les variables LIBSUFFIX et

EXESUFFIX indique les extensions des librairies et des exécutable et prennent les valeurs **lib** et **exe** pour un environnement Windows.

Les définitions existantes dans le fichier Compiler.mak sont implémentées dans tous les autres fichiers Makefile par l'intermédiaire de la directive 'include' du fichier compiler.mak.

```
# ********************* Microsoft C version ************************

BASE = C:\Users\M. NACHAR\Desktop\Nachar

COMPILER     = cl /nologo /D:_MSC_VER=1 @$(BASE)\vc.cfg
LEX          = $(BASE)\exes\flex


MAKE         = nmake/NOLOGO

NAMETAG      = vc

COMPOUT      = /Fo
RESPADD      =
RESPCONT     =
LIBFLAGS     =
LIBTOOL      = lib
LIBOUT       = /OUT:

LINK         =
link /PROFILE /MD
/SUBSYSTEM:CONSOLE /debug /INCREMENTAL:NO /NODEFAULTLIB:msvcrt.lib

LINKOUT      = /OUT:
WINDIS       = $(BASE)\lib\ndif_$(NAMETAG).lib
SOCKLIB      = ws2_32.lib user32.lib
REGLIB       = advapi32.lib

OBJSUFFIX    = obj
LIBSUFFIX    = lib
EXESUFFIX    = exe

DELETE       = -del
#DELETE       = delete
RENAME       = rename
COPY         = !copy
```

Figure D.0.3 Composition du Makefile Compiler.mak dans notre environnement de travail

Adaptation des fichiers de compilation des librairies

Les fichiers de compilation n'ont ni une syntaxe, ni une règle de programmation unique. Cette syntaxe varie selon l'utilitaire permettant leur exécution. Les fichiers de compilation des librairies TMW sont adaptés avec l'utilitaire MAKE de LINUX. Le fournisseur considère que l'implémentation du standard IEC 61850 aura lieu généralement dans des produits basés sur un système d'exploitation embarqué Linux. Toutefois, ce fournisseur laisse la possibilité aux utilisateurs d'effectuer le changement syntaxique de ces fichiers, si un autre environnement de développement de travail est utilisé.

Pour cela, la deuxième étape d'adaptation a consisté à rendre compatible les fichiers de compilation des librairies de fonction en fonction de l'utilitaire d'exécution de ces fichiers de compilation 'NMAKE'.

L'utilitaire Nmake se trouve généralement dans le répertoire de Visual Studio «Microsoft Visual Studio\VC98\Bin ». Ceci permet à l'éditeur de programmation en question d'exécuter les commandes de compilation par un simple clic sur le bouton 'Compile' de l'environnement 'Visual Studio'. Cependant, l'exécution des Makefile de TMW doit être effectuée par une simple commande NMAKE lancée à partir d'une fenêtre de commande (i.e. sans passer par l'environnement de Visual Studio). Pour accomplir cette tâche, une copie de l'exécutable 'Nmake' a été effectuée dans le répertoire system 32 de Windows.

Les modifications de programmation syntaxique, apportées aux fichiers Makefile de TMW pour les rendre compatibles avec l'utilitaire Nmake, peuvent être classées comme suit :

- Modification Syntaxique :
    - Insertion du caractère ' !' avant l'appel aux conditions : ifdef, ifndef, else, endif, include
    - Suppression des caractères ' ;' inséré à la fin de chaque ligne de commande de compilation
    - Remplacement du caractère '**/**' par le caractère '****'
    - Mise entre guillemet des noms des cibles (e.g. PREP = "C:\Users\User\Desktop\Haffar\exes\prep32.exe")
    - Ajout des extensions des exécutable de compilation (e.g. prep est remplacé par prep.exe).
- Modification au niveau de quelques fichiers sources .C constituant les points d'entrées du fichier de compilation (e.g. Remplacement du fichier **eth_lnxp.c** par le fichier **eth_winp.c**)

Librairies de fonction générées

Les cibles qui serviront à l'implémentation du standard IEC 61850 dans la plateforme de Co-Simulation sont les librairies de fonctions TMW.

Suite à la modification des fichiers de compilation et à leur adaptation à notre environnement de travail, nous avons réussi à générer toutes les librairies de fonctions permettant la création des applications IEC 61850.

Le tableau D.1 indique les différentes librairies générées. Ces librairies sont classées par rapport au Makefile qui les a lancées.

Tableau D.1 Librairie de fonctions TMW généré dans un environnement Visual Studio de Windows

maketam.mak tam_vc.lib tamc_vc.lib	**makemms.mak** mms_vc.lib	**makemmsc.mak** mmsc_vc.lib file_vc.ilb
maketask.mak task_vc.lib	**make1006.mak** 1006_vc.lib	**makegoo.mak** goo_vc.lib
makeigoo.mak igoo_vc.lib igos_vc.lib	**makeeth.mak** eth_vc.lib	**makedt.mak** dt_vc.lib
makertg.mak rtg_vc.lib 61850_1.c 61850_1.h	**makestack.mak** stack_vc.lib	**makestack_nc.mak** stack_nc_vc.lib
makecscl.mak cscl_vc.lib	**makecli.mak** cli_vc.lib	**makertgn.mak** rtgn_vc.lib
makertgn2.mak rtgn2_vc.lib	**maketcli.mak** tcli_vc.lib	

Application serveur IEC 61850 sous Windows

Les librairies ainsi générées nous ont servi de point d'entrée au développement des applications IEC 61850. Le premier test, conduit dans notre étude, consistait à utiliser ces librairies de fonctions pour générer les applications des tests fournis par TMW.

En effet, TMW met à disposition un répertoire intitulé exemple. Ce répertoire contient des fichiers Makefile, permettant de prendre en entrée les librairies déjà générées pour produire en sortie (i.e. cibles) des exemples d'applications serveur IEC 61850, tournant sur la machine et utilisant l'interface de communication de cette machine.

La figure D.4 montre le principe de génération des applications de tests IEC 61850 sous Windows.

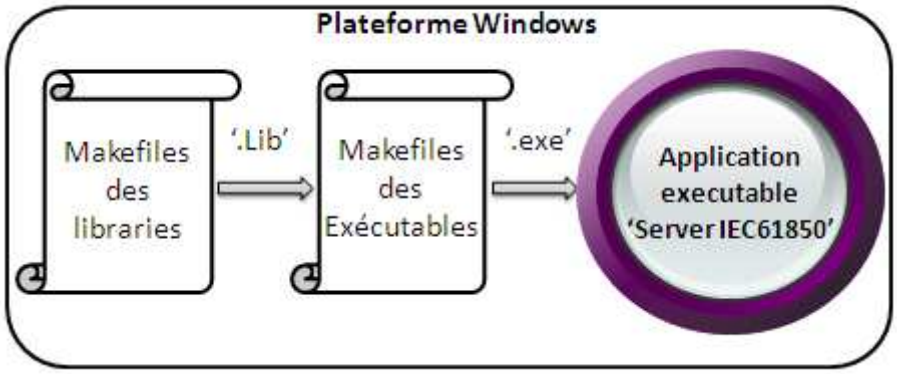


Figure D.0.4 Génération des applications de tests IEC 61850

Toutefois, notre but était de pouvoir implémenter ce standard de communication dans les modèles de communication, eux-mêmes implémentés dans la plateforme de Co-Simulation. Une première implémentation de ces librairies de fonctions a abouti à des erreurs de programmation au niveau de la modélisation et à des échecs au niveau des exécutions de simulation.

Après un débogage informatique détaillé du problème, nous avons constaté qu'il y avait un manque au niveau des étapes d'adaptation des sources protocolaires livrées par TMW. Pour cela, nous avons été amenés à introduire une étape supplémentaire pour accomplir cette mission de modélisation.

Cette étape a été expliquée dans {chapitre 7, paragraphe 7.3.1}, elle a consisté à effectuer des adaptations au niveau des fichiers sources de manière à prendre en compte le stack de communication des modèles de simulation. Nous avons de même montré dans {chapitre 7, paragraphe 7.4}, qu'après avoir effectué ce changement au niveau des fichiers sources TMW, nous avons réussi à modéliser le standard de communication IEC 61850 dans notre plateforme de Co-Simulation et à tester les premiers services de communication implémentés dans nos modèles.

Annexe E :
Formations Professionnelles & Universitaires

Formations universitaires

Pendant mon parcours de thèse, j'ai réalisé deux formations universitaires au sein de l'université Libanaise Faculté de Génie Branche 1. Ces formations ont été présentées sous la forme d'un cours magistrale aux élèves de la deuxième et troisième année cycle ingénieur. Le contenu ainsi que tous les travaux pratiques présentés dans ces deux formations ont été préparés et édités par moi-même. Ce cours a pris en compte les détails de développement d'une application industrielle à base des automates programmables ainsi que le développement des applications de contrôle commande qui permettent de manipuler les signalisations/alarmes et envoyés des signaux de commandes à l'automate via un réseau de communication.

La première formation, intitulée informatique industrielle et système de supervision, a été réalisé au mois d'Octobre 2009 et destinée aux élèves de la dernière année cycle ingénieur. Les objectifs de cette formation sont listés ci-dessous :

- Planifier une application industrielle à base des automates programmables
- Définir les composants matériels adaptés à l'application
- Comprendre les différents langages de programmation des automates et choisir celui le plus adapté à la programmation de l'application
- Construire des architectures de communication à base des automates programmables en passant par la configuration des drivers de communication
- Utilisation des simulateurs des automates programmables
- Développer des applications de surveillance, contrôle et commande des automates programmables
- Construction des architectures de communication industrielle et comprendre les différentes étages de ces architectures et la connexion entre ces étages
- Implémentation et configuration des systèmes de régulation dans les automates programmables et comprendre les effets des différent paramètres d'un régulateur.

Le plan de cette formation est montré ci-dessous :

Chapitre 1 : Introduction

- Les architectures et protocoles de communications industrielles
- Historique des systèmes automatisé à base des relais et introduction à l'informatique industrielle et automates programmable
- Historique des tableaux de contrôle commande à base des voyant et afficheur et introduction des systèmes de supervision

Chapitre 2 : Système de supervision et contrôle/commande

- Création et navigation entre les écrans de supervision
- La base de données des systèmes de supervision et les éditeurs de variables
- Signaux de télécommandes et téléréglage
- Afficheurs de télésignalisations et télémesure
- Création et appel aux fenêtres POPUP
- Manipulation des alarmes et défauts Process et création des groupes et des bandeaux d'alarmes
- Animation des objets synoptique
- Programmation des Scripts de supervision
- Configuration de la couche de communication (protocole, réseau, etc.) des systèmes de supervision
- Travaux dirigés et travaux pratiques assistés

Chapitre 3 : Composants Matériels et Cartes métiers des Automates

- Liste des cartes métiers d'un automate
- Rack d'automate et emplacement spécifiques des cartes
- Techniques du choix des cartes métiers
    - Carte Entrée Tout Ou Rien.
    - Carte Sortie Tout Ou Rien.
    - Carte Entrée/Sortie Analogiques.
- Choix du processeur.
- Choix de la carte d'alimentation
- Principe des téléfastes

Chapitre 5 : Driver de communication et transfert des programmes

- Connexion locale Unitelway à l'automate
- Configuration des drivers de communication
- Connexion distante Ethernet
- Configuration du simulateur de l'automate programmable

Chapitre 6 : Variables

- Variables internes (Organisation Mémoire Automate)
- Variables externe (Cartes Entrées/Sorties – Télécommande/Télésignalisation SCADA)
- Localisation des variables

- Variables élémentaires
- Variables structurées
- Travaux dirigés

Chapitre 7 : Astuces Expérimentales pour la programmation industrielle

- Transcription des variables externe (variable de communication ou entrées sortie physiques) dans un emplacement mémoire interne
- Chevauchement des variables
- Organisation des sections de programme automate
- Création d'une section de simulation de programme
- Gestion du fonctionnement dégradé de l'application et correspondance avec le GEMA

Chapitre 8 : Langage de programmation LADDER

- Intérêt et domaine d'application de ce langage
- Liste d'instructions LADDER
- Règles de programmation
- Construction des Blocs fonctions
- Travaux Pratiques

Chapitre 9 : Langage de programmation GRAPHCET

- Intérêt et domaine d'application de ce langage
- Elément de constitution d'un GRAPHCET
- Règles de programmation
- Définition des actions :
    - Activation d'une section hors répertoire
    - Activation d'une sortie Automate
    - Génération d'une section associée à l'action
- Définition des transitions
- Travaux Pratiques

Chapitre 10 – Régulation PID dans les automates programmables

- De la théorie vers la pratique de régulation
- Effet des paramètres d'un régulateur PID
- Variables associées à un Bloc de régulation PID
- Différents types de sorties d'un régulateur PID intégré dans un automate
- Travaux dirigés et travaux pratiques

Cette formation a été réalisée pendant une période de deux semaines intensives (4 heures par jour). Mise à part les élèves ingénieurs, deux assistants professeurs et des responsables laboratoires ont également assistés à cette formation.

Vue le grand succès de cette formation, les responsables de l'université libanaise m'ont sollicité au mois de février 2010 pour refaire la même formation pour une nouvelle promotion des élèves ingénieurs.

Formation professionnelle et transfert du travail de recherche

A la fin de mon contrat de doctorat, j'ai effectué à l'entreprise Euro System un transfert complet de mon développement de recherche et des connaissances acquises lors de notre travail collaboratif.

Afin de bien mener ce transfert, j'ai préparé une formation professionnelle détaillée pour une durée d'un mois, prenant en compte tous les aspects de R&D que j'ai percuté pendant mon parcours.

J'ai découpé cette formation en 10 travaux pratiques (voir paragraphe suivants) dont chacun a été précédé par un cours illustrant les pré-requis nécessaires pour la réalisation de ces TPs.

Une équipe de travail formée de 3 personnes dont un docteur (Fanny Clavel), un chef de projet (Daniel Baudrand) et un stagiaire Master 2 (Omar Rabi) a été choisie par l'entreprise Euro System afin de suivre ma formation.

Les TPs de cette formation professionnelle, ont été organisés de la manière suivante :

- Les TPs 1, 2, 3 et 4 guident vers la maitrise du logiciel de simulation OpNet Modeler. Ces TPs aboutissent à la création des modèles de simulation basés sur les couches standards TCP/IP. Les points essentiels retenus pendant ces TPs sont :
    - Programmation orientée objet pour les couches de communication des modèles de simulation
    - Identification et manipulation des différents types d'interruption OpNet
    - Interfaçage et cohabitation entre les couches de communications d'un modèle de simulation
    - Création, remplissage, envoi et réception des paquets formatés et non formaté OpNet
    - Programmation des couches applicatives en utilisant des fonctions informatique développées en interne ou externe au logiciel de simulation
    - Programmation des modèles clients et serveurs basés sur les couches standards TCP/IP d'OpNet
    - Création des transactions réseau, enregistrement des statistiques et génération des courbes de performances
- Le TP5 montre les détails de la programmation du module HITL et la configuration nécessaire pour connecter les modèles de simulation à des équipements réels hors l'environnement de la simulation
- Le TP6 montre la manière pour construire des architectures Co-Simulées de base, type (SRS, RSR, etc…), en utilisant le module HITL

- Le TP7 explique l'intérêt et la mise en place du module SITL. Ce TP mène en final à la création des scénarios de communication dynamique par le biais du changement des variables des modèles de simulation en mode d'exécution

- Le TP8 a pour but de programmer le protocole industriel ModBus dans les couches applicatives des modèles de simulation TCP/IP. Le second intérêt de ce TP était de montrer comment utiliser les modules HITL et SITL pour la validation de la modélisation protocolaire

- Le TP9 vise à réaliser des architectures Co-Simulées dans le but d'évaluer leurs performances. Le but essentiel est de montrer l'impact du module HITL sur les retards des transmissions/émissions des messages de communication

- Le TP10 constitue le dernier TP de cette formation professionnel, son but est de montrer la méthodologie pour l'implémentation des services de communication, du standard IEC 61850, dans les couches applicatives des modèles de simulation

TP 1 :
Construction des modèles de base OpNet

But : Maitriser la création des modèles de simulation, les fichiers générés avec un projet de simulation, la prise en main de la programmation orientée objet avec les graphes d'états (étapes/transitions), création des définitions informatique dans le Header Block, maitrise des fonctions de génération des interruptions OpNet.

Pré-requis : formation de base sur OpNet et sur expliquer les interruptions self d'OpNet et les fonctions permettant la génération de ces interruption.

Tâches :

a- Créer un nœud de simulation et associé le à un modèle de nœud constitué d'un seul module OpNet intitulé **'M_Base_OpNet'**. Le module doit être de type 'Processor'

b- Créer un modèle Process intitulé **'M_Base_Process'** et associé le module du nœud de simulation déjà créé à ce modèle de process.

c- Construire un programme Process formé de cinq étapes : **étape 1, étape 2, étape 3, FIN, INCONNU**. La transition entre les étapes est réalisée de la manière suivante. Initialement, le Process doit se trouver dans l'étape 1. Le passage de l'étape 1 vers l'étape 2 est conditionné par la transition **'Trans1'**. Cette dernière est caractérisée par la réception d'une interruption de code égale à 'après 5 secondes du passage dans l'étape 1. Le passage de l'étape 2 vers l'étape 3 est non conditionnée, il est accomplit à la fin du traitement de l'étape 2. Le passage de l'étape 3 vers l'étape 1 est conditionné par la transition **'Trans3'**. Cette dernière est caractérisée par la réception d'une interruption interne de code égal à '3' après 5 secondes de rentré dans cette étape. Le cheminement (étape 1 – étape 2 – étape 3) doit se réaliser au bout de deux itérations Process. Les transitions doivent être définit dans le Header Block d'OpNet.

d- Dans la troisième itération, l'étape 1 doit guider le process vers une étape indiquant la réception d'une interruption inconnue pour revenir après à la même étape.

e- Dans la quatrième itération, l'étape 1 doit guider le process vers l'étape fin ou le process affiche l'état qu'il est à sa fin et attente la fin de la simulation ou il met une fin à la simulation.

f- Exporter le projet vers une autre machine de simulation et créer un nouveau process intitulé **'M_Base_Process_V2'** et un nouveau nœud intitulé **'M_Base_OpNet_V2'** en modifiant le process actuel par l'élimination de l'étape **'INCONNU'**. Monter l'effet d'un tel changement sur le comportement de la simulation.

Expérimentation assistée 2 :
Construction des modèles client/serveur TCP/IP

But : Création des modèles client serveur TCP/IP, Identification et manipuler des interruptions Stream OpNet, Utilisation des fonctions appartenant à des fichiers sources externe OpNet, création des paquets formaté et envoi/réception des paquets de simulation via le stack de communication des modèles OpNet.

Pré-requis : formation sur les fonctions de création des paquets, sur les librairies d'établissement de connexions, les interruptions Stream.

Tâches :

a- Créer un autre nœud de simulation, associer le à un modèle de nœud intitulé **'serveur TCP/IP'** et composé d'un module applicatif vide connecté au modules standard TCP/IP livré par OpNet. Associer le module applicatif d'OpNet au modèle Process intitulé par **'serveur TCP/IP Process'**. Configurer ce nœud à l'@ IP 10.10.10.2 et programmer son Process de manière à effectuer une connexion passive sur le port 200.

b- Créer un autre nœud de simulation, associer le à un modèle de nœud intitulé **'client TCP/IP'** et composé d'un module applicatif vide connecté au modules standard TCP/IP livré par OpNet. Associer le module applicatif d'OpNet au modèle Process intitulé par **'client TCP/IP Process'**. Configurer le client à l'@ IP 10.10.10.2 et **p**rogrammer son Process applicatif de manière à générer une demande de connexion active vers un serveur simulé d'@IP= 10.10.10.3 à l'écoute sur un le port 200.

c- Construire une architecture réseau simulée connectant les deux modèles de simulation créés et déterminer en utilisant les fonctions de détection des types et code d'interruption les résultats des connexions passives et actives (connexion active réussite, interruption Stream du TCP vers l'application doit être envoyée avec un code égal à 2).

d- Modifier le Process applicatif du client de manière à envoyer après l'établissement de connexion avec le serveur un paquet formaté OpNet. Modifier le Process du serveur de manière à afficher le contenu des paquets applicatifs envoyés par le modèle client.

e- Renvoyer un autre paquet formaté par le modèle client au serveur sans informer la couche transport que l'application serveur est prête à recevoir un nouveau (remarque : le paquet ne sera pas reçu par l'application serveur, mettre à jour le process serveur pour permettre cette réception)

Expérimentation assistée 3 :
Génération des transactions et calcul des performances

But : Manipulation envoi et réception des paquets non formatés ; création d'une transaction client/serveur ; enregistrement des statistique et création des courbes de performances.

Pré-requis : formation sur les fonctions de manipulation des paquets non formatés ; la création, l'enregistrement et la génération des statistiques.

Tâches :

a- Créer un nouveau nœud de simulation intitulé **'client TCP/IP_V2'** héritant toutes les caractéristiques que **'client TCP/IP'**. Effectuer un changement au niveau du process de ce modèle intitulée **'client TCP/IP Process_V2'** de manière à envoyer un paquet non formaté contenant une structure formée d'un tableau de deux bytes et un tableau de deux Int.

b- Changer le nœud de simulation serveur pour créer un autre intitulé **'serveur TCP/IP_V2'** de manière à ce que lorsqu'il détecte une réception d'un paquet de type non formaté, il transite vers une étape intitulé **'Pk_Unf_Rcv'** dans laquelle il affiche le contenu de ce paquet. Le process applicatif de ce serveur est intitulé **'serveur TCP/IP Process_V2'**

c- Programmer une transaction réseau de manière à ce que si les deux bytes du paquet envoyés par le client sont égaux à '1', le serveur répond au client en renvoyant le même paquet. A la réception de la réponse du serveur, le client affichera le contenu de la réponse envoyée.

d- Créer une variable statique dans le modèle client intitulée **'délai transaction'**. Le calcul de cette variable se fera dans le modèle client de manière à ce qu'elle soit égale à la différence entre le temps d'envoi de la réponse du serveur et le temps d'envoi de la requête client (NB : Ajuster votre échelle, de manière à prendre en compte que cette variable va être dans l'ordre des microsecondes).

e- Créer une statistique intitulée 'Transaction Delay' associée à un groupe intitulé 'Performance'. Cette statistique doit être écrite à chaque envoi d'une requête client, elle doit être attribuée à la valeur de la variable **'délai transaction'** calculé dans l'étape précédente.

f- Démarrer une simulation de deux minutes dans laquelle le client est configuré à envoyer des requêtes périodique à chaque 10 secondes. Ces requêtes doivent alternées entre un envoi transactionnel et un envoi normal (n'attendant pas de retour de la part du serveur). Générer les performances d'une telle simulation et analyser les résultats obtenus.

Expérimentation assistée 4 :
Attributs des modèles de simulation

But : Création des attributs des modèles de simulation; utilisation des attributs dans les process des couches applicatives.

Pré-requis : formation sur la généralisation de la programmation et sur les fonctions OpNet permettant la manipulation des attributs des modèles de simulation.

Tâches :

NB : Toutes les trames envoyées dans ce TP doivent être de type transactionnelle

a- Modifier le modèle client de manière à ce qu'il construise des paquets formaté dont la taille est spécifié en tant qu'un attribut à ce modèle

b- Modifier le Process du client de manière à ce qu'il prend en compte la périodicité de l'envoi des trames à partir des attributs du modèle.

c- Lancer plusieurs scénarios en changeant la périodicité de l'envoi des trames client (conclusion : attributs simplifient la tâche de programmation des process mais ne permettent pas un changement dynamique en mode d'exécution de la simulation)

Expérimentation assistée 5 :
Modèle OpNet en Co-Simulation

But : Maitrise du module HITL de la plateforme de Co-Simulation ; maitrise du type d'échange des paquets via le module HITL, réalisation des Co-Simulation en connectant les modèles de simulation déjà créés à des équipements réel via HITL.

Pré-requis : formation sur la construction du module HITL de la plateforme ; les configurations nécessaires à mettre en place pour démarrer une Co-Simulation.

Tâches :

a- Créer une application serveur exécutable (VB) permettant d'afficher dans un Textbox le contenu des paquets applicatifs envoyés par un client distant.

b- Connecter cette application à distance au modèle **'client TCP/IP_V2'** par l'intermédiaire du module HITL de la plateforme de Co-Simulation en faisant la configuration nécessaire au niveau de la plateforme de Co-Simulation et vérifier l'échange de communication entre le modèle de simulation et l'application distante.

c- Rajouter le modèle **'client TCP/IP'** permettant l'envoie des paquets formatés mais connecté le cette fois ci au serveur réel via la Co-Simulation.

d- Changer le port de communication au niveau du modèle client et de l'application serveur et appliquer un filtre au niveau du numéro de port pour limiter les paquets échangés avec l'environnement de la Co-Simulation.

e- Conclure que sur le fonctionnement du module HITL

   a. Sur quel port de communication fonctionne ? (accepte n'importe quel échange de communication sur n'importe quel port)
   b. Comment appliquer le filtrage de paquets ? (le filtrage doit se faire au niveau des attributs de la passerelle HITL)
   c. Quels sont les types de paquets acceptés à ce jour par le module HITL ? (les paquets échangés au niveau de la plateforme de Co-Simulation doivent être de type non formatés pour le bon fonctionnement de cette dernière)
   d. Au niveau de quel profil de communication ces paquets doivent communiquer ? (profil TCP/IP)
   e. Perspective pour l'utilisation du module HITL pour une Co-Simulation IEC 61850 (modifier la librairie HITL de manière à échanger les paquets UDP et les paquets Ethernet)

Expérimentation assistée 6 :

Architectures de base SRS et RSR construite par l'intermédiaire de la plateforme de Co-Simulation

But : Maitrise d'une architecture de Co-Simulation connectant deux modèles de simulation par l'intermédiaire d'un réseau réel (double Co-Simulation à effectuer); Construction d'une architecture de Co-Simulation connectant deux équipement réel en passant par un réseau de communication virtuel construit dans OpNet

Pré-requis : formation sur la construction du module HITL de la plateforme ; les configurations nécessaires à mettre en place pour démarrer une Co-Simulation.

Tâches :

a- Connecter l'application serveur Windows à l'application cliente Windows par l'intermédiaire de la plateforme de Co-Simulation. Construire un réseau virtuel dans l'environnement de Co-Simulation et vérifier l'échange entre les deux équipements par l'intermédiaire de cet environnement. Identifier la première architecture de base construite par l'intermédiaire de la plateforme de Co-Simulation (RSR)

b- Connecter le modèle client au modèle serveur (à choisir si version 1 ou version 2) par l'intermédiaire d'un réseau de communication réel. Identifier la deuxième architecture de base SRS créée au travers notre plateforme de Co-Simulation.

c- Modifier l'architecture SRS de manière à introduire des modèles réseaux OpNet et vérifier le bon déroulement de la Co-Simulation.

Expérimentation assistée 7 :
Intérêt et mise en place du module SITL

But : Connexion d'une application externe SITL à la plateforme de Co-Simulation dans le but de la création des scénarios de communication dynamique et le changement au niveau des variables Process en mode d'exécution d'une Co-Simulation.

Pré-requis : Formation sur la librairie SITL et implémentation dans la plateforme de Co-Simulation, implémentation de la librairie ODBC pour l'échange entre l'application SITL et OpNet, utilisation des fonctions de la librairie SITL dans le process OpNet.

Tâches :

a- Démarrer une Co-Simulation $S_cR_rR_s$ connectant le modèle client à l'application serveur et envoyer un paquet non formaté de deux bytes avec une période de 5 secondes. Relancer la simulation en modifiant les valeurs de deux bytes. Qu'est ce que vous remarquez ? (Arrêt de la Co-Simulation pour changer les valeurs des bytes envoyés)

b- Créer une simple table SQL contenant trois champs Integer 'INT1', 'INT2' et 'PERIODE' à échanger entre une application de contrôle et le modèle de simulation client. Modifier la librairie SITL de manière à prendre en compte ces trois valeurs de la base de données. Créer une application Windows permettant de contrôler ces trois variables SQL.

c- Modifier le process de l'application cliente de manière d'aller lire les deux bytes de la base de donnée en mode d'exécution de la simulation.

d- Créer par l'intermédiaire de cette solution SITL un scénario de communication dynamique en modifiant la période de génération des échanges de communication entre le client simulé et le serveur réel. Cette période doit être écrite en fonction de la valeur PERIODE dans la table SQL.

   a. Discussion au niveau des scénarios périodiques et évènementiels des signaux Goose

e- Effectuer un changement au niveau de l'application et le module SITL et du Process client de manière à générer des communications évènementielles.

Expérimentation assistée 8 :

Modélisation ModBus & Utilisation de la plateforme de Co-Simulation pour la validation de la modélisation protocolaire

But : Prise en main du travail de développements des modèles ModBus ; maitrise de l'utilisation de la plateforme de Co-Simulation pour la validation protocolaire.

Pré-requis : Connaissance protocole industriel ModBus ; Maitrise HITL ; maitrise du module SITL et de l'application SITL créer pour le contrôle commande des modèles ModBus.

Tâches :

a- Comprendre le développement du module client ModBus ainsi que les différentes parties du process de ce modèle y compris celle connectant le modèle client au module SITL et la connexion avec le module HITL.

b- Démarrer une Co-Simulation SRR dans laquelle le modèle client sera connecté à un serveur réel du marché (choisissez le serveur Unity Pro). En utilisant l'application SITL, créer plusieurs scénarios de communication en envoyant toutes les fonctionnalités de communication ModBus modélisé dans ce client. Valider la modélisation du protocole ModBus dans le logiciel de Simulation en utilisant la plateforme de Co-Simulation (HITL).

c- Comprendre le développement du modèle serveur ModBus ainsi que les différentes connexions entre le serveur et les modules de la plateforme de Co-Simulation

d- Démarrer une Co-Simulation SRR dans laquelle le modèle serveur sera connecté à un client réel du marché (choisissez le PCVUE). Configurer l'application de supervision de manière à envoyer les différentes fonctionnalités de communication implémentée dans le modèle serveur. Valider la modélisation du serveur par l'intermédiaire de la Co-Simulation effectuée.

Expérimentation assistée 9 :

Evaluation de performances des architectures Co-Simulées ModBus

But : Montrer l'intérêt de la plateforme de Co-Simulation pour l'évaluation de performance des architectures Co-Simulées. Validation de performances délivrés par la plateforme de Co-Simulation. Points à prendre en compte lors de future évaluation de performances des architectures Co-Simulées (Partie soulignée est le but d'un stage).

Pré-requis : Maitrise des modèles clients et serveur ModBus ; maitrise de l'application de contrôle commande du modèle client.

Tâches :

a- En utilisant les modèles client et serveur ModBus déjà développé et en utilisant l'application SITL de contrôle commande du modèle client, construire une seule Co-Simulation contenant deux architectures la première de type $S_cR_rR_s$ et la deuxième de type $S_cR_rS_s$. L'élément commun entre les deux architectures sont le modèle client et les équipements réseaux. Le changement au niveau des deux architectures est au niveau du modèle serveur. En utilisant l'application SITL divisée la Co-Simulation en une partie attaquant le modèle serveur et une autre partie attaquant le serveur réel (NB : le serveur réel doit être de type cyclique).

b- Générer les performances d'une telle Co-Simulation et valider l'effet du modèle serveur en le comparant à un serveur réel pour une architecture Co-Simulée. Conclure en réalisant d'autre Co-Simulation que la différence entre les performances des deux architecture vient du module HITL de la plateforme de Co-Simulation.

c- Réaliser une architecture complètement réelle RRR et une autre complètement simulée SSS. Valider les résultats délivré par le logiciel de simulation

d- Conclure qu'une architecture complète virtuelle donne des résultats donne des résultats de performance similaire à une architecture réelle, le passage vers la Co-Simulation ajoute des retards imposée par la translation et la prise en compte des messages par le module HITL

Expérimentation assistée 10 :
Modélisation et validation du standard IEC 61850

But : Montrer les premières modélisation effectuées pour le stack de communication IEC 61850. Montrer l'architecture permettant la validation de cette modélisation.

Pré-requis : Maitrise des logiciels de configurations et de supervision IEC 61850 livré par le fournisseur TMW, maitrise des Makfile, maitrise de l'environnement de développement de TMW.

Tâches :

a- Expliquer les différentes étapes pour la modélisation du standard IEC 61850

b- Mettre en place l'architecture de validation de la modélisation IEC 61850 en particulier les service de Read MMS et le service d'autodiscovery

Références Bibliographiques

Adamiak M., Redfern M., [1998], communication systems for protective relaying, *in* 'IEEE computer applications in power', vol.11, pp.14 – 18.

Alessandria E., Seno L., Vitturi S. [2007], *in* '7th IFAC International Conference on Fieldbuses and Networks in Industrial and Embedded Systems', session 8, pp. 391 – 398.

Ali I., Thomas M-S. [2008], Substation Communication Networks Architecture, *in* 'Joint International Conference on Power System Technology and IEEE Power India Conference, 2008'. POWERCON 2008, pp. 1 - 8.

Apostolov A. [2010a], IEC 61850 Substation Configuration Language and Its Impact on the Engineering of Distribution Substation Systems, *in* 'International Congress on Electricity Distribution', paper no. 31.

Apostolov A., Vendiver B. [2010b], Scheduled Versus Event-Driven Testing of Distribution Protection IEDs, *in* 'International Congress on Electricity Distribution' paper 32.

Apostolov A., Brunner C., Clinard K. [2003], Use of IEC 61850 object models for power system quality/security data exchange, *in* 'Quality and Security of Electric Power Delivery Systems, CIGRE/IEEE PES International Symposium', pp. 155 – 164.

Apostolov A. [2003], Disturbance Recording in IEC 61850 Substation Automation Systems, *in* 'Transmission and Distribution Conference and Exhibition, IEEE PES', pp. 921 - 926.

Apostolov A., Vendiver B. [2004], Functional Testing of IEC 61850 Based IEDs and Systems, *in* 'Power Systems Conference and Exposition', IEEE PES, pp. 640 – 645.

Atienza E. [2010], Testing and Troubleshooting IEC 61850 GOOSE-Based Control and Protection Schemes, *in* '63rd Annual Conference for Protective Relay Engineers', pp 1 – 7.

Avizienis A., Laprie J-C., Randell B., Landwehr C. [2004], Basic Concepts and Taxonomy of Dependable and Secure Computing, *in* 'IEEE Transactions On Ddependable And Secure Computing', Vol. 1, No. 1, pp. 11 – 33.

Barger Pavol [2003], Evaluation et validation de la fiabilité et de la disponibilité des systèmes d'automatisation à intelligence distribuée en phase dynamique, thèse de doctorat, université henry poincarré, Nancy 1.

Bayart M. [2004], Instrumentation intelligents, systèmes automatisés de production à intelligence distribuée - Habilitation à Diriger des Recherches, USTL, Lille, 21 décembre 1994.

Beugin Julie [2006], Contribution à l'évaluation de la sécurité des systèmes complexes de transport guidé, thèse de doctorat, Présentée à l'université de valenciennes et du hainaut-cambrésis, UMR CNRS 8530, UVHC Le Mont Houy 59313 VALENCIENNES Cedex 9.

Bruandet G., Angays P., Haffar M., Saxena P-K. [2010], IEC 61850 Tutorial, *in* 'Petroleum and Chemical Industry Technical Conference', PCIC 2010 Europe.

Caetano C., Pernes M. [2007], Introducing IEC61850 in Distribution Substations Automation Systems, *in* 'PowerGrid Europe T&D – Unifying Europe', 26-28 June 2007, Madrid.

Cauffriez Laurent [2005], Méthodes et modèles pour l'évaluation de la sûreté de fonctionnement de systèmes automatisés complexes, habilitation à diriger des recherches, université de Valenciennes et du Hainaut-Cambrésis.

Chen J., Ren Y., Gao X., Jin Y. [2008], The research on conformance testing platform of numerical substation, *in* 'China International Conference on Electricity Distribution', pp. 1 - 5.

Chenghong T., Bin S., Guo H., Hanguang P., Xiaojun L., Ji Z. [2008], Design Of Electrical Anti-misoperation System Based on IEC61850 Digital Substation, *in* 'International Conference on Electrical Engineering', paper no. O-204.

Chotin E., Benoit E., Foulloy L. [1998], An approach for cooperation in distributed architectures, *in* 9th symposium on information control in manufacturing, INCOM'98/IFAC, Nancy, 24-26 juin 1998.

Cockburn B.F. [1995], Deterministic Tests for Detecting Scrambled Pattern-Sensitive Faults in RAMs, *in* 'IEEE International Workshop on Memory Technology, Design and testing'.

Engler F., Kern T-L., Andersson L., Kruimer B., Schimmel G., Schwarz K. [2004], IEC 61850 based digital communicationas interface to the primary equipment, *in* 'International Council On Large Electric Systems', session 2004, B3-205.

Goraj M. [2006], Substation Automation System Based On IEC 61850 Architecture, *in* 'Proceeding of the 15th International Conference on Power System Protection', PSP 2006.

Guo J., Xiang W., Wang S. [2007], Reinforce Networking Theory with OPNET Simulation, *in* 'Journal of Information Technology Education', Vol.6, pp. 215 – 226.

Haffar M., Thiriet J-M, Savary E. [2007], Modeling of substation architecture implementing IEC 61850 protocol and solving interlocking problems, *in* '7th IFAC International Conference on Fieldbuses & Networks in Industrial & Embedded systems', Toulouse, France (November 7-9, 2007), pp. 291-294.

Haffar M., Khan Z-H., Thiriet J-M, Savary E. [2008], An extension to IEC 61850 for solving selectivity problem in electrical substations, *in* '23rd IAR Workshop on Advanced Control and Diagnosis 2008', Coventry, 27-28 November 2008, pp . 362-367.

Haffar M., Thiriet J-M [2010a], Software and hardware in the loop component for an IEC 61850 Co-Simulation platform, *in* 'Proceedings of the International Multiconference on Computer Science and Information Technology', pp. 817–823, 18-20 October 2010, Wisla, Poland. ISBN 978-83-60810-27-9 ISSN 1896-7094.

Haffar M., Thiriet J-M., Nachar M. [2010b], A Hardware in the Loop module in an IEC61850 Co-Simulation Platform for advanced substation automation system tests, *in* 'IEEE International Energy Conference and Exhibition', EnergyCon 2010, 18-22 Décembre 2010, Manama, Bahrain, paper 1569328789.

Haffar M., Savary E., Baudrand D. [2010c], IEC61850 Co-Simulation platform: an advanced tool for Substation Automation System design architecture, *in* 'IEEE International Energy Conference and Exhibition', EnergyCon 2010, 18-22 Décembre 2010, Manama, Bahrain, paper 1569315305.

Haffar M., Thiriet J-M, Kabbara S. [2010d], Validation of Hardware and Software in the Loop add-ons simulation modules, *in* 'Journées de la Section Automatique Démonstrateurs en Automatique 30 novembre', 1 décembre 2010 à Angers.

Haffar M., Thiriet J-M, Mechref K., Ziade H. [2010e], Transforming your computer center to an Emulated Industrial Laboratory, *in* 4ème congrès international francophone de mécanique avancée, Alep (Syrie), 19-21 avril 2010.

Hakala-Ranta A., Rintamaki O., Starck J. [2005], UTILIZING POSSIBILITIES OF rsc 61850 AND GOOSE, *in* 'International Conference and Exhibition on electricity Distribution', session 3, pp. 1 – 4, paper no. 0741.

Hasan M-S., Yu H., Griffiths A., Yang T-C. [2008], Co-simulation framework for Networked Control Systems over multi-hop mobile ad-hoc networks, *in* '17th World Congress, The International Federation of Automatic Control', IFAC 2008, pp. 12552 – 12557.

Hnatyshin V., Simmons M. [2004], Practical Methodology for Development and Deployment of Standard Applications Process Models using OPNET Modeler, *in* 'OPNET's annual technology conference'.

Hoga C., Wong G. [2005a], Communication according to IEC 61850 – Reliability at high speed, *in* 'International Conference And Exhibition on electricity Distribution', pp. 1 – 4.

Hoga C., Wong G. [2005b], Utilities and industries gain benefits as IEC 61850-implementation gains speed, *in* 'IEEE Power Engineering Society Inaugural Conference and Exposition in Africa', pp. 176 – 179.

Hossenlopp L. [2005], INCREASING SUBSTATION AUTOMATION STANDARDIZATION LEVEL, *in* 'International Conference And Exhibition on electricity Distribution', session 1, pp. 1 – 5.

Hutchinson G.P [1966], Interlocking in large electricity-supply substations - a fundamental approach, *in* 'Proceedings of the Institution of Electrical Engineers', Vol. 113.

Ito H., Kaneda K., Hamamatsu K., Tanaka T., Nara K. [2008], Dependability Evaluation of Substation Automation System with Redundancy, *in* '12th WSEAS International Conference on SYSTEMS'.

Irith P., Sudhakar M.R. [1993], 3-Weight Pseudo-Random Test Generation Based on a Deterministic Test Set for Combinational and Sequential Circuits, *in* 'IEEE Transactions on computer aided design of integrated system', Vol.12, No.7.

Kakanakov N., Shopov M., Spasov G., Hristev H., Performance Evaluation of Switched Ethernet as Communication Media in Controller Networks [2007], *in* 'International Conference on Computer Systems and Technologies', session IIIA, paper no. 8.

Kalappa N., Acton K., Antolovic M., Mantri S., Parrott J., Luntz J., Moyne J., Tilbury D. [2006], *in* 'IEEE Conference on Emerging Technologies and Factory Automation', pp. 1061 - 1064.

Lloret P., Velásquez J-L., Molas-Balada L., Villafáfila R., Sumper A., Galceran-Arellano S. [2007], IEC 61850 as a flexible tool for electrical systems monitoring, *in* 'International Conference Electrical Power Quality And Utilisation', EPQU 2007.

Nguyen-Dinh N., Kim G-S., Lee H-H [2007], A study on GOOSE communication based on IEC 61850 using MMS ease lite, *in* 'International Conference on Control, Automation and Systems', ICCAS '07, pp. 1873 - 1877.

Ozansoy C-R., Zayegh A., Kalam A. [2002], Communications For Substation Automation And Integration, *in* 'Australasian Universities Power Engineering Conference', AUPEC 2010.

Pereira N., Tovar E., Pinho L-M. [2004], INDEPTH: Timeliness Assessment of Ethernet/IP-based Systems, *in* 'The IEEE Computer Society's 12th Annual International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems', MASCOTS 2004, pp. 192 - 201.

Ren J., Kezunovic M. [2010], Modeling and Simulation Tools for Teaching Protective Relaying Design and Application for the Smart Grid, *in* 'Modern Electric Power Systems 2010', MEPS'10, Wroclaw, Poland, paper no. 52.

Rietmann P., Reindhart P. [2006], Applying IEC 61850 to Substation Automation Systems, *in* 'Proceeding of the 15th International Conference on Power System Protection', PSP 2006.

Robert M. [1992], Contribution à l'évolution de l'instrumentation industrielle : capteurs intelligents et détections de défauts, Habilitation à Diriger des Recherches, UHP, Nancy, 29 septembre 1992.

Robert J., Georges J-P, Rondeau E., Divoux T. [2010], Analyse de performances de protocoles temps-réel basés sur Ethernet, in 'Conférence Internationale Francophone d'Automatique'.

Santos J., Silva N., Rodrigues P., Rodrigues A., Marsh D., Gomes F., Mota Pinto C., Blanquet A., Carrapatoso A. [2009], Electric Grid Versus Data Network Architectures And Standards Smartgrid As Plug&Play, *in* 'International Conference And Exhibition on electricity Distribution', session 3, paper no. 0912.

Saxena P., Meyer E., Roy P., Saeed N. [2010], Experience In Design, Engineering, Testing & Implementation Of Electrical Systems For A Large GAS Processing Plant, *in* 'Petroleum and Chemical Industry Technical Conference Record, PCIC Europe 2010, pp. 1-8.

Shokooh S., Khandelwal T., Shokooh F. [2005], Intelligent Load Shedding Need for a Fast and Optimal Solution, *in* 'Petroleum and Chemical Industry Technical Conference Record, PCIC Europe 2005.

Shookoh F., Dai J-J., Shookooh S., Tastet J., Castro H., Khandelwal T., Donner G. [2005], An Intelligent Load Shedding (ILS) System Application in a Large Industrial Facility, *in* 'Fourtieth IAS Annual Meeting. Conference Record of the Industry Applications Conference', pp. 417 – 425.

Sidhu T-S., Yin Y. [2007], Modelling and Simulation for Performance Evaluation of IEC61850-Based Substation Communication Systems, *in* 'IEEE Transactions On Power Delivery, Vol. 22, No. 3, pp. 1482 – 1489.

Sidhu T-S., Injeti S., Kanabar M-G, Parikh P-P [2010], Packet Scheduling of GOOSE Messages in IEC 61850 based Substation Intelligent Electronic Devices (IEDs), *in* 'IEEE Power and Energy Society General Meeting'.

Skendzic V., Guzmia A. [2006], Enhancing Power System Automation Through The Use Of Real-Time Ethernet, *in* 'Power Systems Conference: Advanced Metering, Protection, Control, Communication, and Distributed Resources', PS '06, pp. 480 – 495.

Tan J-C., Zhang C., Bo Z-Q. [2008], The Importance of IEC 61850 Interoperability Testing, *in* '43rd International Universities Power Engineering Conference', UPEC 2008, pp. 1 – 5.

Tan J-C., Green V., Ciufo J. [2009], Testing IEC 61850 Based Multi-vendor Substation Automation Systems for Interoperability, *in* 'Power Systems Conference and Exposition', PSCE '09. IEEE/PES, pp. 1- 5.

Thomas M-S., Ali I. [2010], Reliable, Fast, and Deterministic Substation Communication Network Architecture and its Performance Simulation, *in* 'IEEE Transactions On Power Delivery, Vol. 25, No. 4, pp. 2364 – 2370.

Udren E-A., Dolezilek D. [2007a], IEC 61850: role of conformance testing in successful integration, *in* 'International Council On Large Electric Systems', S3-7: IEC 61850 influencing the design of secondary systems.

Udren E., Strabbing W., Dolezilek D. [2007b], IEC 61850: Role of Conformance Testing in Successful Integration, *in* 'International Conference And Exhibition on electricity Distribution', session 3, paper no. 0006.

Vaurio J.K [2002], Treatment of General Dependencies in System Fault-Tree and Risk Analysis, *in* 'IEEE Transaction on Reliability', Vol. 51, No. 3, SEPTEMBER 2002

Xyngi I., Popov M. [2010], IEC61850 overview - where protection meets communication, *in* ' International Conference on Developments in Power System Protection', DSPS 2010.

Yu L., Beck R-T. [1983], Reliability and Availability Studies for Industrial Power System Analysis, *in* 'IEEE Transactions On Industry Applications, Vol. IA-19, No. 6, pp. 968 – 974.

Zhao S., Choi M., Lee S, Han S [2009], Latency Evaluation and Analysis of IEC 61850 Based SAS Communication Network, *in* 'International Conference on Advanced Power System Automation and Protection'.

Zhang W., Zhang W [2005], Priority scheduling in switched industrial Ethernet, *in* 'American Control Conference', Vol. 5, pp. 3366 – 3370.

Zhang X., Liu Q., Wang S. [2008], Real-time Communication Based on MMS+TCP/IP+ Ethernet for Embedded CNC, *in* '4th International Conference on Wireless Communications, Networking and Mobile Computing', WiCOM '08, pp. 1 - 5.

Normes et Documentations Techniques

Normes

INTERNATIONAL STANDARD IEC 61850-1, Communication Networks and Systems in Substations, Part 1: Introduction and Overview, First edition 2003-04, www.iec.ch

INTERNATIONAL STANDARD IEC 61850-5, Communication Networks and Systems in Substations, Part 6: Configuration description language for communication in electrical substations related to IEDs, First edition 2004-03, www.iec.ch

INTERNATIONAL STANDARD IEC 61850-6, Communication Networks and Systems in Substations, Part 5: Communication Requirements for Functions and Device Models, First edition 2003-07, www.iec.ch

INTERNATIONAL STANDARD IEC 61850-7-1, Communication Networks and Systems in Substations, Part 7-1: Basic communication structure for substation and feeder equipment – Principles and models, First edition 2003-07, www.iec.ch

INTERNATIONAL STANDARD IEC 61850-7-2, Communication Networks and Systems in Substations, Part 7-2: Basic communication structure for substations and feeder equipment - Abstract communication service interface (ACSI), www.iec.ch

INTERNATIONAL STANDARD IEC 61850-8-1, Communication networks and systems in substations – Part 8-1: Specific Communication System Mapping (SCSM) – Mappings to MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3, First edition 2004-05, www.iec.ch

INTERNATIONAL STANDARD IEC 61850-9-1, Communication networks and systems in substations – Part 9-1: Specific Communication System Mapping (SCSM) – Sampled values over serial unidirectional multidrop point-to-point link, First edition 2003-05, www.iec.ch

INTERNATIONAL STANDARD IEC 61850-10, Communication networks and systems in substations – Part 10: Conformance testing, First edition 2005-05, www.iec.ch

Documentations Techniques

Adamiak M., Patterson R., Melcher J., Inter and Intra Substation Communications: Requirements and Solutions, http://store.gedigitalenergy.com/faq/documents/369/ger-3976.pdf.

Altuve H-J., Thompson M-J, Mooney J., Advances in Breaker-Failure Protection, www.wseas.us/e-library/conferences/2008/crete/Systems/sys2-59.pdf

BTIB, Gestion Technique fédérée GTF, http://solar.energy.free.fr/logiciel_automatisme/Lonworks_LonMark.pdf

E3 Group of Spanish Electricity companies for studies on IEC 61850 [2010], minimum common specification for substation protection and control equipment in accordance with the IEC 61850 standard, www.nettedautomation.com/.../61850/E3_IEC61850_Specification_Document_20100609.pdf

Guide de protection, Electrical network protection, *in* 'www.schneider-electric.com'.

Guide de conception des réseaux électriques industriel, *in* 'www.schneider-electric.com'.

Guide of the IEEE Power Engineering Society, Approved Draft IEEE Guide for Breaker Failure Protection of Power Circuit Breakers, www.pes-psrc.org/.../PC%2037119%20draf7%2005-16-05.pdf.

IEC TC56 WG3 Dependability Management [2009], Concept Of Dependability, www.sars.org.uk/forms/dependability.pdf.

IEEE PSRC H6: Special Report [2005], Application Considerations of IEC 61850/UCA 2 for Substation Ethernet Local Area Network Communication for Protection and Control, http://www.pes-psrc.org/Reports/H6Paper-App%20Consider%20of%20IEC61850&UCA_072205_083105.pdf

IEEE-SA Technical Report on Utility Communications Architecture (UCATM) Version 2.0, *in* IEEE--SA TR 1550—1999.

Janssen M-D., Koreman C.G.A., Substation Components Plug And Play Instead Of Pluf And Pray The impact of IEC 61850, www.nettedautomation.com/.../Substation_Components_Plug_and_Play_instead_of_Plug_and_Pray.PDF

Le Men Newron System, Intégration intéropérable en technologie LonWorks: bâtiment Infogrames, www.metaproductique.com/infogrames/fichiers/InfogrammesGlobal.pdf.

Sautriau F. [1990], Protection des réseaux par le Système de Sélectivité Logique, in 'cahier technique 2 Schneider',www.schneider-electric.com/documents/.../electrotechnique/.../ct002.pdf.

Tutorial Makefile, Makefile-Club des décideurs et professionnels en informatique. *Développez.com*. [En ligne] 2005. http://gl.developpez.com/tutoriel/outil/makefile/.

Udren E-A., Dolezilek D., IEC 61850: Role of Conformance Testing in Successful Integration, www.selinc.com.

WSCC Telecommunications, Relay Work Groups, Communications Systems Performance Guide Fors Protective Relaying Applications, www.wecc.biz/.../Communication%20System%20Relay%20Guide.pdf.